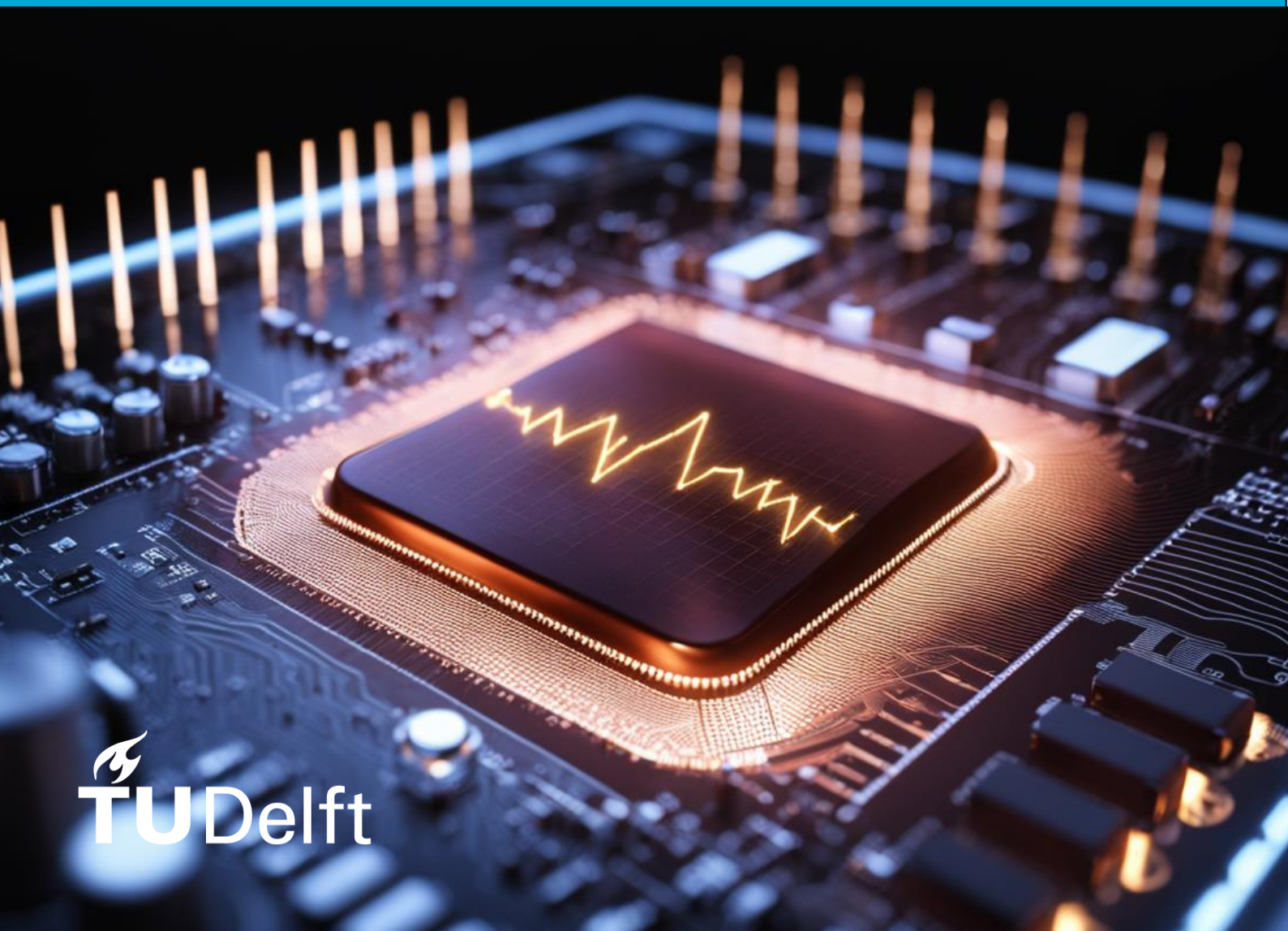
MSc in Microelectronics

Energy-efficient Spike Encoding for ECG Arrhythmia Classification

Yingzhou Dong

Student number: 5471192

2023



TU Delft

MSc in Microelectronics

Energy-efficient Spike Encoding for ECG Arrhythmia Classification

Yingzhou Dong

September 2023

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Microelectronics

Yingzhou Dong: *Energy-efficient Spike Encoding for ECG Arrhythmia Classification* (2023)
An electronic version of this thesis is available at https://repository.tudelft.nl/.

The work in this thesis was carried out in the:

Quantum & Computer Engineering
Delft University of Technology

Supervisors:	Dr. Rajendra Bishnoi	QCE, TU Delft
	Prof.dr. Said Hamdioui	QCE, TU Delft
Committee member:	Dr. Chang Gao	ME, TU Delft

Abstract

Cardiovascular diseases (CVDs) are the top cause of death worldwide, and their diagnosis can be quickly and painlessly achieved through Electrocardiogram (ECG). The diagnosis of electrocardiogram has gradually evolved from manual diagnosis by doctors to one that can be realized using Artificial Intelligence (AI). Early AI still required manual extraction of features for ECG classification, but later Deep Neural Network (DNN) could automatically extract features during the learning process. With this technology, people can monitor heart movements in real-time through wearable devices. If there are any abnormalities, they can seek medical treatment in time to prevent death from sudden severe heart disease. However, for the wearable device, the energy consumption per classification by the traditional AI is so high that a limited battery cannot work for a long time.

To address this issue, this thesis adopts Spiking Neural Network (SNN) to do classification and implement the inference on the hardware. Compared with traditional Artificial Neural Network (ANN), SNN is highly energy efficient. According to the needs of SNN and its event-driven characteristics, the multi-threshold-based encoding scheme is proposed, which encodes the heartbeat into 54 spikes on average with less information loss. The SNN model is trained by ANN-SNN conversion with an accuracy of 97.42%. After RTL coding, synthesis, and back-end implementation, the chip with encoding and inference functions achieves an energy consumption of only 57.88 nJ per heartbeat classification.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Dr. Rajendra Bishnoi, for his patient guidance and support throughout my thesis journey. He has been an extraordinary supervisor, providing me with meticulous and gentle guidance at every step of the way. I am truly fortunate to have had such a dedicated supervisor. I am also deeply thankful to Prof. Said Hamdioui for his insightful analysis and suggestions. Their contributions have been instrumental in shaping my thesis.

Sumit Diware deserves special recognition for his technical expertise and enthusiastic help. At vital moments, he provided a lot of learning materials, and when I was stuck, he pointed out the problem with his rich experience.

I also extend my heartfelt thanks to my friends, who were always there with encouragement. Your company made the journey all the more enjoyable and memorable.

I am deeply thankful to all those who played a part, directly or indirectly, in my academic and personal growth.

Delft, September 2023

Yingzhou Dong

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

Cardiovascular diseases (CVDs) encompass a collection of disorders that impact the heart and blood vessels, particularly the arteries and veins. In 2020, cardiovascular diseases (CVDs) were documented as the leading global cause of mortality [40, 4]. Cardiovascular disease includes ischaemic heart disease and stroke. According to the World Health Organization (WHO), as shown in Figure 1.1, ischaemic heart disease and stroke account for 16% and 11% of the world's total deaths respectively [2]. It can be seen from the figure, that each of these are far outnumbered compared to the other causes of death. CVDs are responsible for 27% of total deaths. The world heart federations has forecasted that the number of deaths from heart diseases may reach a staggering 22.3 million in 2030 [3]. High blood pressure, high cholesterol, unhealthy diet, physical inactivity, and more are the primary risk factors for CVDs.

Cardiac arrhythmia, one of the most prevalent issues in cardiology, frequently accompanies the majority of cardiovascular disease patients. It refers to the issue of the pace or pattern of heartbeat, which performs as too quick, too slow, or irregular rhythm. In modern medical science, detection techniques for heart disease have undergone significant advances. Electrocardiogram (ECG) is the most commonly used and very effective cardiac arrhythmia detection method. The ECG records the electrical activity of the heart and provides medical professionals with rich information that can be used to diagnose abnormal heart rhythms and study problems with the heart's electrical conduction.

Early detection of cardiac abnormalities provides individuals with the opportunity to get appropriate treatment that potentially prevents the patient from entering a dangerous situation. At the same time, early intervention also helps reduce healthcare costs, as simpler treatment options tend to be more effective earlier in the disease. As a result, detecting cardiac arrhythmia in time by monitoring ECG in real-time is very meaningful.

A long time ago, ECG diagnosis relied on doctors. Patients must first go to the hospital to obtain ECG graphics through professional equipment and then submit them to doctors for diagnosis. Due to the doctors' enormous workloads, the diagnosis may be made many days after the submission of the ECG recording. This is inconvenient and delays the time for the patient to discover the abnormality, which might lead to the emergence of severe heart conditions.

To solve the above problems, currently, ECG can be obtained through wearable devices [15, 7, 47]. Patients can complete the acquisition of ECG in their daily lives, and with the real-time analysis system, the acquired ECG can be diagnosed in time. In the first stage, automatic computer-based ECG classification can do investigation using Support Vector Machine (SVM) [53], random forest [36], and Naive Bayes [44] as pattern recognition techniques.

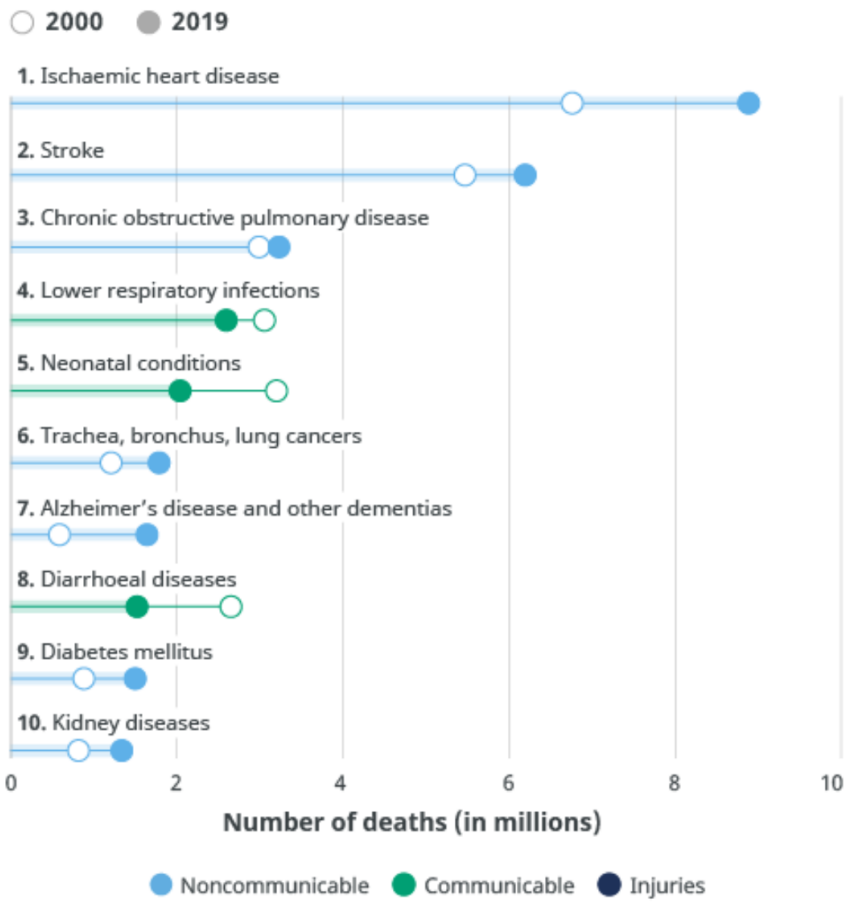


Figure 1.1: The leading causes of death based on data from WHO, 2019 and 2020 [2]

However, the drawback of those techniques is that manual preprocessing is required to extract the ECG signal features before processing in the AI model. The feature extraction has high precision requirements; otherwise, it damages the accuracy of classification.

In recent years, many projects have adopted Deep Neural Networks (DNN) to address this problem. The neural network automatically learns features through training to make accurate inferences about heartbeat arrhythmia. Using such automation, the step of manually extracting information is eliminated. Many works have tried different architectures for ECG arrhythmia detection. For example, Wang et al [55] and Hannun et al [26] use Convolutional Neural Network (CNN), Chauhan et al [12] uses Long Short Term Memory Network (LSTM), He et al [27] uses Temporal Convolutional Network (TCN), etc. Even though some of them achieve good accuracy through sophisticated neural network model design, the energy consumption is high in hardware [55, 32, 48].

For wearable devices, there is a requirement that the power should be as low as possible. Because the battery volume of wearable devices is limited, frequent charging will affect the user experience. It will reduce customers' desire to use it. It is also possible that frequent charging may cause patients to occasionally forget to put it back on, resulting in missed

monitoring. Reduced power consumption extends the device's post-charging usage duration, which is a goal of wearable devices. How to further reduce AI energy consumption has become the focus.

Recently, Spiking Neural Network (SNN) began to receive wide attention because of its high energy efficiency. SNN neurons are closer to biological neurons. Biological cells have evolved over the years and can complete biological activities using very little energy. For example, the chip ODIN achieves 15 nJ/inference for image classification[21].

As for ECG classification using SNN, although some of them achieve very low energy consumption, the accuracy drops a lot [39, 57], which is only about 90%. Since arrhythmia detection is related to health, high accuracy is the most important requirement. The reduced accuracy means that some danger signs are not detected in time, which will cause patients to miss early treatment. Moreover, if the normal heartbeat is analyzed as severe abnormality, the patient will go to the hospital in panic, wasting time and medical resources. Some other works have high accuracy, but some encoding has low energy efficiency [6, 14]. Most of them need preprocessing in software [6, 41]. Chu et al's [14] work claims that their encoding can be implemented in hardware, but they have not achieved it yet, so the outcome is unknown.

In view of the above situation, in this thesis, we have implemented the complete inference flow in hardware, including the encoding module, achieved high accuracy, and fully explored the low energy consumption of SNN.

1.2 Problem statement

- As a medical device, it aims to monitor the electrocardiogram (ECG) in real-time to detect arrhythmia in time. When abnormal heartbeats are detected, it guides patients to seek medical treatment in time and reduces the death caused by acute severe heart disease. Therefore, the accuracy is highly important.
- This chip is embedded into the wearable device. Due to the limited volume of the battery, the power of the chip should be as low as possible. In terms of the energy consumption per classification, the lower, the better, as well.
- Encoding is very important in SNN because it influences the calculation time and accuracy. The hardware implementation of the encoding module is ignored by previous works; preprocessing needs to be done externally, like CPU. It makes the total classification time long and the system on chip complex, accompanied by energy loss when transferring data.

1.3 Contributions

This thesis presents a complete solution of ECG classification in a chip using SNN with high accuracy and ultra-low energy consumption. We make full use of the low energy consumption characteristics of SNN to prolong the use time of the device after a single charge and improve the user experience. The key contributions of this thesis are as follows:

- **Multi-threshold-based encoding scheme, a new encoding scheme for ECG signal**
  As a critical step in SNN, the new encoding scheme saves most of the original ECG information, which maintains the high accuracy of classification. Meanwhile, it produces a small number of spikes and is area-saving in hardware, which leads to ultra-low energy consumption in Hardware calculation.
- **Design space exploration for an optimal SNN model**
  The baseline ANN model and ANN-SNN conversion tool are carefully selected. During the training in software, CrossEntropyLoss and Scheduler are adopted to acquire the optimal SNN model for high accuracy. Moreover, weights are quantized for prototyping.
- **Hardware prototyping to reproduce inference in an energy-efficient way**
  A chip is designed with an encoding module and a classifier inference module on it. A special design is made to resolve the issue of increased calculation time due to repeated calculations caused by the conversion tool. After RTL design, verification is conducted to ensure that the inference result and process values are exactly the same as those in the software. Next, synthesis and back-end layout and routing are carried out to generate the chip layout.

The proposed encoding scheme encodes each heartbeat into 64 spikes on average. The designed chip achieves 97.42% accuracy in classifying 11 types of heartbeats. On average, the energy consumption for each classification is 57.88 nJ. Compared to the state-of-art [41], it saves 80.67% energy consumption per classification.

1.4 Organization

Except for this chapter, chapters are organized as follows:

- Chapter 2 explains the background information of cardiac arrhythmia and the basic conception of Spiking Neural Network, including the working principle, the encoding scheme, and training methods.
- Chapter 3 discusses three ECG encoding schemes of state-of-art works. Their advantages and disadvantages are analyzed.
- Chapter 4 first shows the new encoding scheme. Then, how the SNN model is obtained is revealed. After that, the implementation in hardware from RTL design to physical layout is present.
- Chapter 5 begins with the simulation setups, followed by the simulation results in software and hardware.
- Chapter 6 summarises the thesis. Some suggestions for future work are given.

2 Background

This Chapter introduces the background of this thesis, including cardiac arrhythmia and the Spiking Neural Network, which are helpful for later design explanation.

2.1 Cardiac arrhythmia

2.1.1 Basic concept

The ECG signal, electrocardiogram, is a common medical diagnostic tool that has been used extensively for decades as a test of cardiac function. It detects different movements of the heart by collecting changes in electrical signals through electrodes attached to the surface of the skin.

Figure 2.1 shows an ECG graph of a normal heartbeat. The x-axis represents time, and the y-axis represents detected voltage. The ECG signal for one heartbeat contains three parts:

- The P wave: it represents the atria depolarization.
- The QRS complex: it represents ventricles depolarization.
- The T wave: it represents the repolarization of the ventricles.

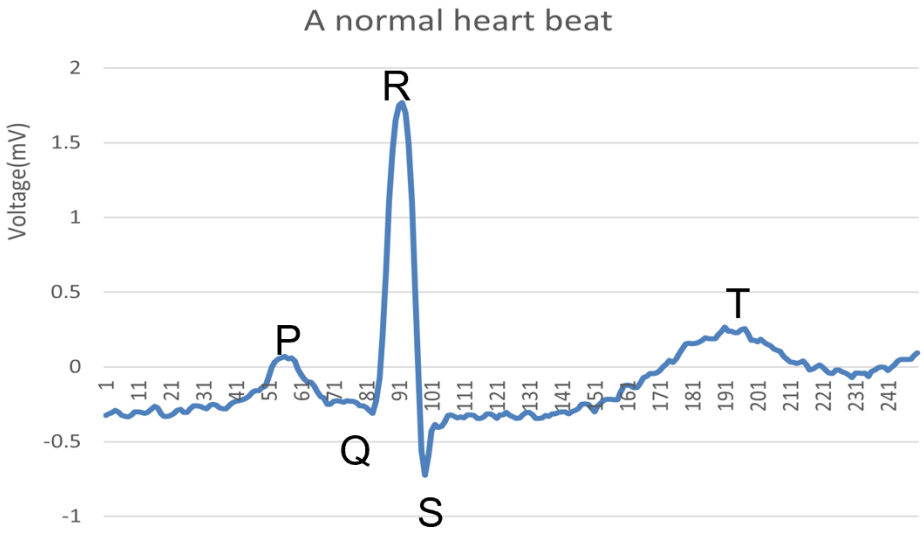


Figure 2.1: The ECG pattern for a normal heartbeat from MIT-BIH Arrhythmia Database [42]

Every normal heartbeat has an orderly depolarization process. In terms of the performance of ECG, P wave, QRS complex, and T wave show appropriate shapes and time intervals [51, 37, 8]. If the shape of the ECG pattern changes, it indicates that the movement of the heart has been changed. QRS complex gathers the main information of the heartbeat and is a key factor in identifying arrhythmias, such as amplitude, distance, duration, angle, slope, and a few other characteristics. Some pathological changes have common features in ECG. Learning the features of different arrhythmia on ECG performance can help doctors or machines to make quick diagnoses. Those features are universal for all human beings.

2.1.2 Arrhythmia classes

In this thesis, the database of ECG is the MIT-BIH Arrhythmia Database [42], which can be downloaded from PhysioNet [23]. This database is collected from Beth Israel Deaconess Medical Center, which was named Boston's Beth Israel Hospital in 1975. Scientists recorded the ambulatory ECG signal of 47 subjects. Then, they digitalized the data to an 11-bit resolution and 360 samples per second. It has been widely used in published ECG arrhythmia classification studies and thus serves as an excellent reference for comparison with other studies.

For heartbeats used in this work, they are segmented into 250 samples for each heartbeat, and the amplitude is 10 mV. In total, there are approximately 110,000 heartbeats labeled with types.

The MIT-BIH Arrhythmia classifies heartbeats into 15 types, shown in Table 2.1. According to the Association for the Advancement of Medical Instrumentation (AAMI) [52], the 15 types of arrhythmia in the MIT-BIH database can be grouped into five superclasses, also shown in Table 2.1.

AAMI Class	MIT-BIH Arrhythmia Class
Normal (N)	Normal Beat (N)
	Left Bundle Branch Block Beat (L)
	Right Bundle Branch Block Beat (R)
	Atrial Escape Beat (e)
	Nodal (junctional) Escape Beat (j)
Supraventricular Ectopic Beat (SVEB)	Atrial Premature Beat (A)
	Aberrated Atrial Premature Beat (a)
	Nodal (junctional) Premature Beat (J)
	Supraventricular Premature Beat (S)
Fusion Beat (F)	Fusion of Ventricular and Normal Beat (F)
Ventrciular Ectopic Beat (VEB)	Premature Ventricular Contraction (V)
	Ventricular Escape Beat (E)
Unknown Beat (Q)	Paced Beat (/)
	Fusion of Paced and Normal Beat (f)
	Unclassifiable Beat (Q)

Table 2.1: AAMI classes [52] and MIT-BIH Arrhythmia Classes [42] for ECG arrhythmia

2.2 Spiking Neural Network architecture

2.2.1 Working principle

Compared to traditional Artificial Neural Network (ANN) neurons, SNN neurons are more biologically plausible. There are some types of SNN neurons proposed, such as the Leaky-Integrate-and-Fire (LIF) [33], the Hodgkin–Huxley model (HH model) [28], the Izhikevich model [29]. Due to its computational simplicity, the LIF model is now one of the most commonly used models for analyzing the behavior of the nervous system [18, 30]. For further simplification, sometimes the leaky part calculation is discarded, and only the integrate and fire are reserved; this creates a new LIF subtype called the Integrate-and-Fire (IF) neuron. The schematic of the IF neuron is shown in Figure 2.2.

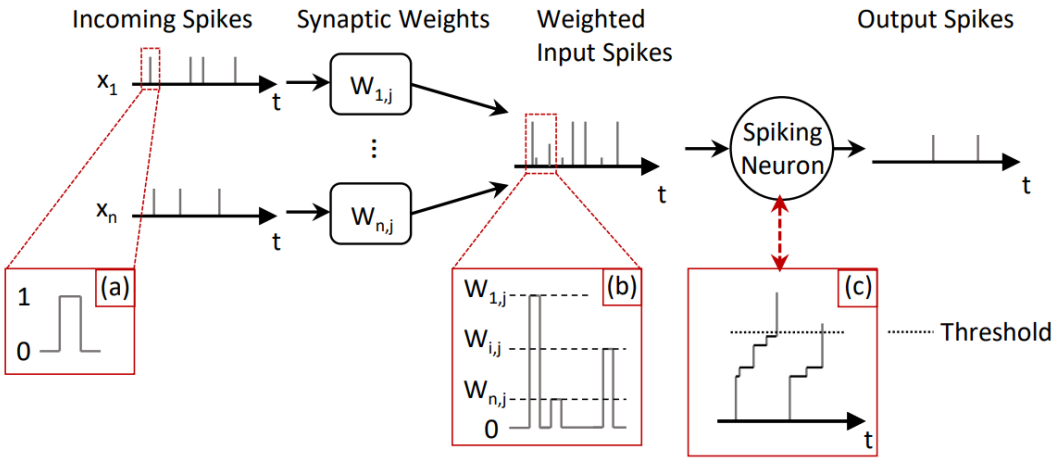


Figure 2.2: The illustration of SNN IF neuron[10]. (a) A standard spike; (b) Weighted input spikes; (c) The membrane potential of a spiking neuron and its threshold (the dotted line).

As shown in the schematic, the input and output of IF neurons are spikes. After the input spikes are received by the synapse, they will be multiplied by a certain weight to become weighted input spikes. Those weighted input spikes get integrated into the membrane potential of the spiking neuron. Once the membrane potential is large enough and exceeds the threshold, the IF neuron fires and sends a spike to the synapse of the next neuron.

The working method of an IF neuron is very close to that of a biological neuron. Neurotransmitters (spikes) are transmitted between neurons, received by neurons through synapses, and information (the membrane potential) is gradually accumulated. When a certain level (threshold) is reached, this neuron is triggered (fires), and neurotransmitters are sent to the next neuron (output spikes).

2.2.2 Training methods

There are three ways to train the SNN model: spike-time dependent plasticity (STDP), ANN-SNN conversion, and Backpropagation [46]. Their characteristics are discussed as follows:

- **Spike-time dependent plasticity (STDP)**
  This is a method of unsupervised learning based on dependencies between presynaptic and postsynaptic spikes [22, 9, 50]. Like other unsupervised learning, the accuracy of STDP is not very high, usually under 90%.

- **ANN-SNN conversion**
  ANN-SNN conversion is supervised learning, so the accuracy is high compared to unsupervised learning. First, an ANN model is required; then, by using some conversion techniques, the ANN model is transformed into an SNN model. This training method is suitable for the existing ANN model, and the training of the ANN model is now very mature and common. In the process of converting ANN to SNN, accuracy may be lost. The loss of accuracy may be a bit large before. With the deepening of its research, the loss of this accuracy is gradually getting smaller. Now, there are various methods to reduce the accuracy loss, such as threshold rescaling [49], soft-reset [25], weight normalization [19], and threshold shift [17].

- **Backpropagation**
  Backpropagation [58] is also a supervised learning method with high accuracy. Because the membrane potential of SNN neurons is a step function, which is nondifferentiable, the Backpropagation method suffers from a problem that the gradient cannot be calculated during backpropagation. Subsequently, some researchers proposed the surrogate function so that the gradient can be calculated in SNN backpropagation [43, 34, 56], but the number of layers of the neural network is limited because if the number of layers is large, gradient explosion will occur. In order to solve the problem of gradient explosion, more special processing needs to be used, making the SNN training method of Backpropagation particularly complicated.

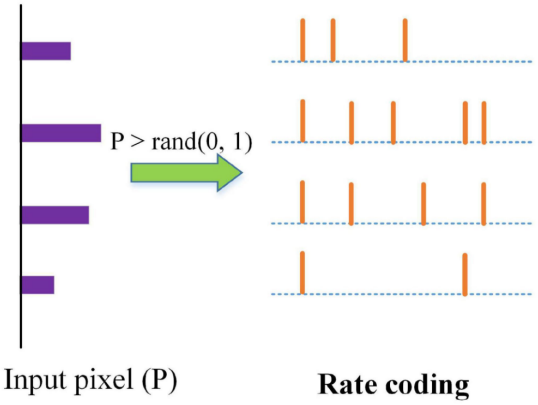
2.2.3 Encoding methods

Since the inputs of the SNN model are spikes, the original data need encoding to be converted into spikes, which is a 1-bit high-level signal in electrical signals. The majority of SNN works use rate encoding, and temporal encoding is adopted because some scientists believe in biological neurons, some information is encoded in a temporal way [54].

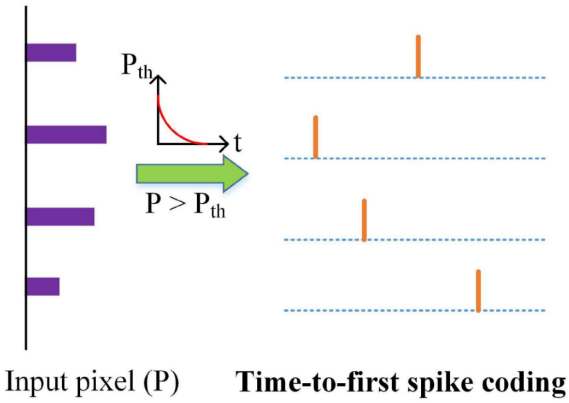
- **Rate encoding**
  Rate encoding is a method of randomly generating a series of spikes, but the probability of spike generation is proportional to the value of the original data, as shown in Figure 2.3a. For example, Poisson encoding is a kind of rate encoding, and the probability of a random spike generation satisfies the Poisson distribution.

- **Temporal encoding**
  The time when Temporal encoding generates a spike is related to the value of the original data. For example, in Time-to-First Spike encoding (TTFS) [45], as shown in Figure 2.3b, the larger the value, the earlier the spike will be generated. Conversely, the earlier the spike represents, the larger the original data.

Besides rate encoding and temporal encoding, there are also some other ways to encode the data, like phase coding [31], burst coding[24], and so on.



(a) Rate encoding example: the probability to generate random spikes is proportional to original value. The larger value generates more spikes.



(b) Temporal coding example: Time-to-first spike coding. The larger value generates a spike at an earlier time. Each data is encoded into one spike at different time.

Figure 2.3: Examples of different encoding schemes with original data and encoded data [24]

2.2.4 Connection layers of Artificial Neural Network

The spiking neural network has the same connection structure as the traditional ANN. Between the input layer, output layer, and the hidden layer, those layers could be connected by the convolution layers, fully connected layers, max-pooling layers, and so on. The big difference between SNN and traditional ANN lies in the activation layer. Commonly used activation layers of Traditional ANN are sigmoid function and Rectified Linear Unit (ReLu) function. For SNN, as introduced in Section 2.2.1, the common activation layer is LIF or IF.

Figure 2.4 shows the structure of one fully connected layer. Each output neuron is connected to the input neuron by a different weight.

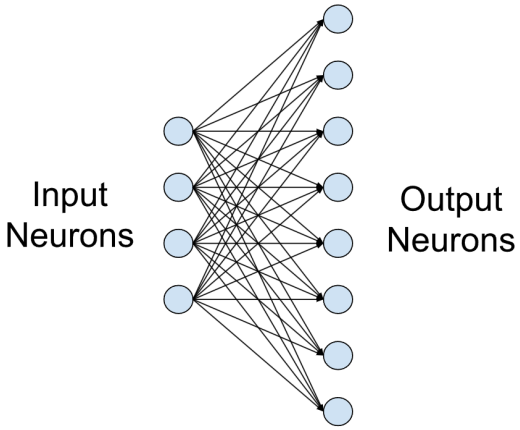


Figure 2.4: The fully connected layer[5]

The value of j_{th} neuron is calculated in the following way:

$$y_j = \sum_i D_i \times W_{ij} + Bias \tag{2.1}$$

Where i represents the serial number of the previous neuron. D and W represent the data and weight, respectively.

2.2.5 The concept of time-step

Time-step refers to a discrete unit of time used to simulate dynamic neural activity. In the mathematic model, it refers to the number of input vectors. Traditional ANN uses continuous-valued activation; there is usually one input vector, and the inference result is based on the output of that input vector. Events in SNN are represented as discrete spikes or events that occur at specific points in time. Therefore, SNN needs multiple timesteps or vectors of input and accumulates their output to do inference [38, 13, 35].

For example, if the model adopts Poisson encoding, which is rate encoding, to encode the values 0.2 and 0.8, only one vector of random number generation can not tell the difference. Value 0.8 could generate no spike in one time-step, while value 0.2 could generate one spike. Only after multiple time-steps, when random spikes are generated following Poisson distribution, can it be clearly seen that the number of spikes generated by value 0.8 is 4 times that of 0.2. Therefore, the neural network can acquire enough information to make the right decision. Figure 2.5 is another example of time-steps. As the time-step increases, the encoded image is more close to the original image.

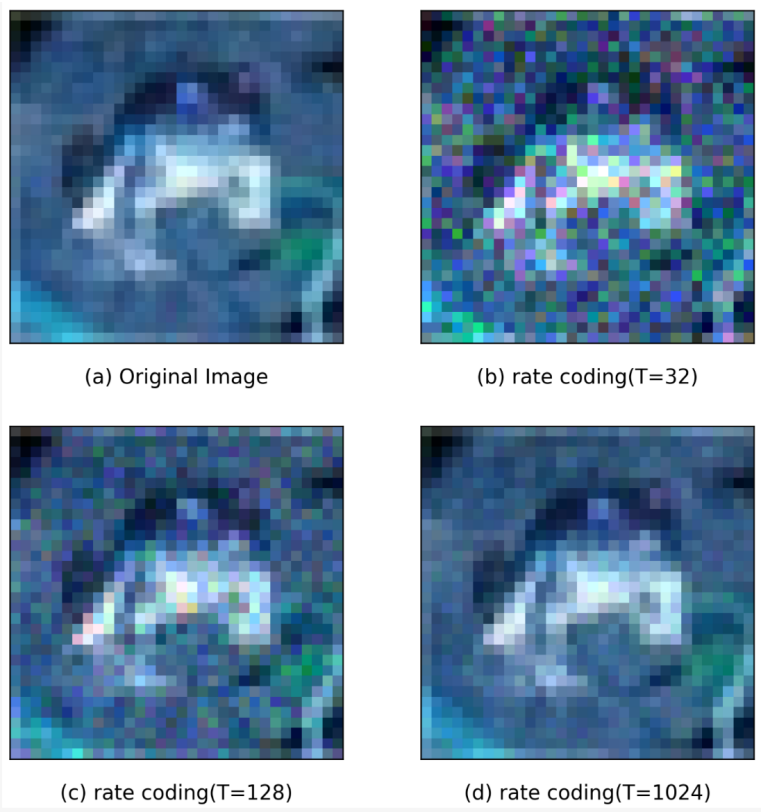


Figure 2.5: An example of rate coding effect in different time-steps [35]

3 Related Work

Encoding performs an essential role in spiking neural networks. It converts the raw data into spikes. And it profoundly affects the computational performance of SNN. A large number of spikes leads to a large calculation load. If the number of timesteps is large, the calculation time will be long, leading to significant energy consumption as well. Moreover, compared with raw data, the encoded data suffers from information loss, reflected in the decline in accuracy. As for ECG classification using SNN, most of them use Poisson encoding. In this chapter, Poisson encoding, Dual-purpose binary encoding, and LC sampling will be introduced in detail, and their advantages and disadvantages will be analyzed.

3.1 Poisson encoding for ECG signal

Poisson encoding is one kind of rate encoding. The possibility of spike generation fits Poisson distribution. The larger the value, the higher the probability of generating spikes, and after accumulating a certain timestep, the greater the total number of spikes generated. Because its accumulated spike number is proportional to the original value, it can be directly applied to the model converted from ANN. It is easy for the existing neural network to be trained with the original data. Amirshahi et al archives an SNN model for ECG classification in analogue circuits [6]. The accuracy is 97.9% and it consumes 1.78 μJ per classification.

To test the Poisson encoding's effect, we have done an experiment. The original data of Poisson encoding should be within the range of 0 and 1, but the ECG data from the MIT-BIH dataset ranges from about -5 mV to 5 mV. So, the first step is to normalize the original data. Then, we use the normalized data to train the ANN model. In this process, normalization has made the accuracy drop from 98.30% to 96.24%.

Next, the ANN model is converted to an SNN model. The input is encoded through Poisson coding. The SNN model is tested with encoded spikes, and the relationship between the accuracy of the SNN model and timestep is shown in Figure 3.1. The orange line is the accuracy of the ANN model; the blue line shows the change in the SNN accuracy with timesteps increasing. From the figure, we can see, in the beginning, the accuracy of the SNN model is very low, lower than 40%. It approaches the accuracy of the ANN model until 1000 timesteps. Such a large number of timesteps will result in a very long calculation time. This also means that the total energy consumption will be large. Even though the SNN consumes less power than the ANN, the final effect may not be very good if the calculation time is too long.

We also have done an experiment on the MNist dataset [16]; the number of timesteps is much smaller. The reason for the large number of timesteps for the ECG dataset with Poisson encoding might be that diagnosis of the ECG signal requires high precision. The longer the spikes accumulate, the closer the shape of encoded data is to the shape of the original data.

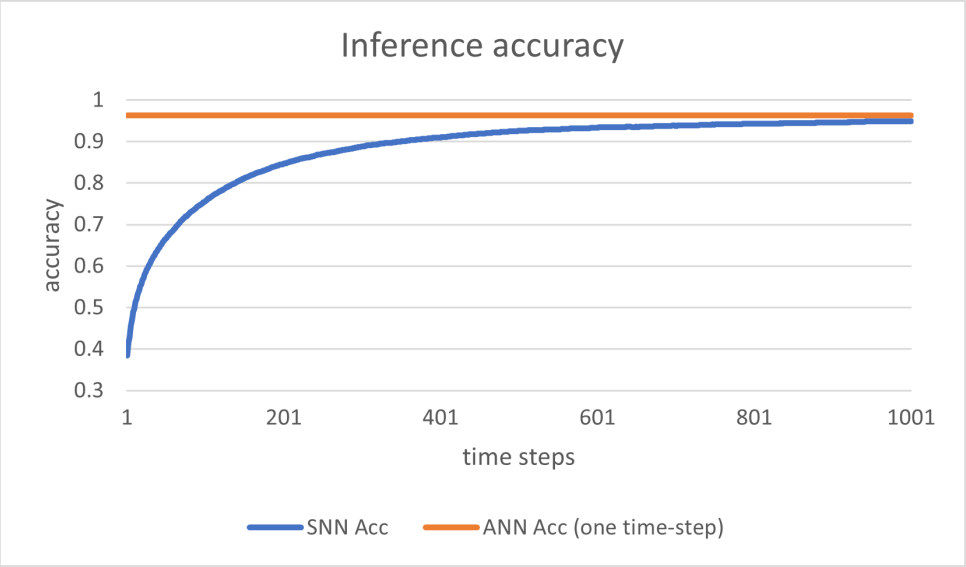


Figure 3.1: The change of SNN accuracy with the time-step increasing, and the relative relation between it and the accuracy of ANN

According to our experiment, the Poisson encoding is not ideal for ECG classification, not only because of the accuracy loss during original data normalization but also because of the large timesteps it needs to take, leading to extremely long calculation time.

3.2 Dual-purpose binary encoding scheme

Mao et al [41] propose a specific encoding scheme for ECG signal, named dual-purpose binary encoding scheme. This encoding scheme encodes each heartbeat into 96 bits in total, of which the amount is tiny. The detailed steps are shown in Figure 3.2.

The dual-purpose binary encoding scheme works in the following steps:

- Step 1: Record the time interval from the current R peak to the previous R peak and subsequent R peak, respectively. These two time intervals occupy 10 bit, respectively. Another 2 bits are utilized for the change of the R peak intervals.
- Step 2: Discard the flat part of ECG signals.
- Step 3: Encode the rest of the changing part with ”1” representing increase and ”0” representing decrease.
- Step 4, combine the above encoded data in a special order as shown in Figure 3.2.

The advantage of this encoding scheme is the data width, and the number of spikes is small. Therefore, the energy consumption will be small with this encoding scheme. Meanwhile, their SNN model achieves an accuracy of 98.6% with this encoding scheme, which means most of the important information is preserved during encoding. However, this encoding scheme is relatively complex. The original data needs to be preprocessed in software. If

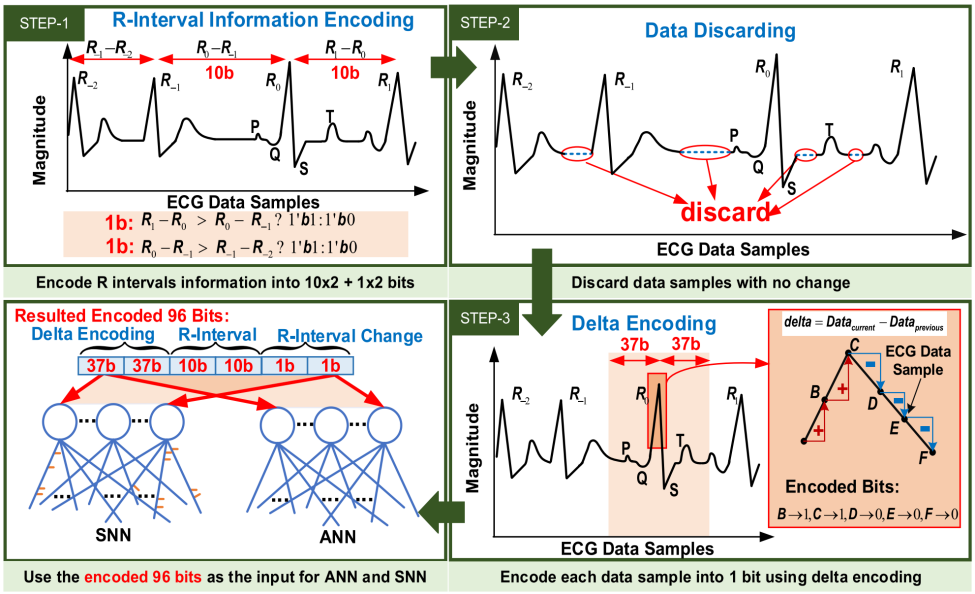


Figure 3.2: The specific steps of dual-purpose binary encoding proposed by Mao et al [41]

we look at the overall system, complex coding is done in software and then passed to the hardware chip of SNN for reasoning. This coding preprocess still consumes energy and requires an additional software processor. For wearable devices, it is better that energy consumption can be further reduced.

3.3 LC sampling

Chu et al [14] proposed the LC sampling. This belongs to the threshold-based encoding. The threshold-based encoding is very suitable for continuous signals like ECG. In the top figure of Figure 3.3, the red horizontal lines are the threshold. When the ECG signal increases above a threshold, a rising spike will be generated; conversely, if the ECG signal falls below a threshold, a falling spike will be generated. If there are neither rising spikes nor falling spikes for a while, it means the signal remains almost unchanged. Dense spikes mean the signal changes rapidly during this time (see QRS peak in Figure 3.3).

Although their SNN model has an accuracy of 98.22% and the LC sampling can be realized by two comparators in the circuit, this encoding scheme is not simple in hardware implementation and has low energy efficiency to some extent. One reason is that the spikes are sampled from two leads; each rising or falling channel contains 480 data points, and half of the encoded data is reused in time step 2 in Figure 3.3(c). Therefore, the encoded data of each heartbeat is 2880 bits. This large number of encoded data will result in a considerable calculation time. Another reason is the threshold gap is 0.1 mV. In hardware implementation, the calculation is in binary. If we convert it to a binary number, it is:

$$(0.1)_{10} = (0.0001100110011001100110011001100110011001100110011001101)_2$$

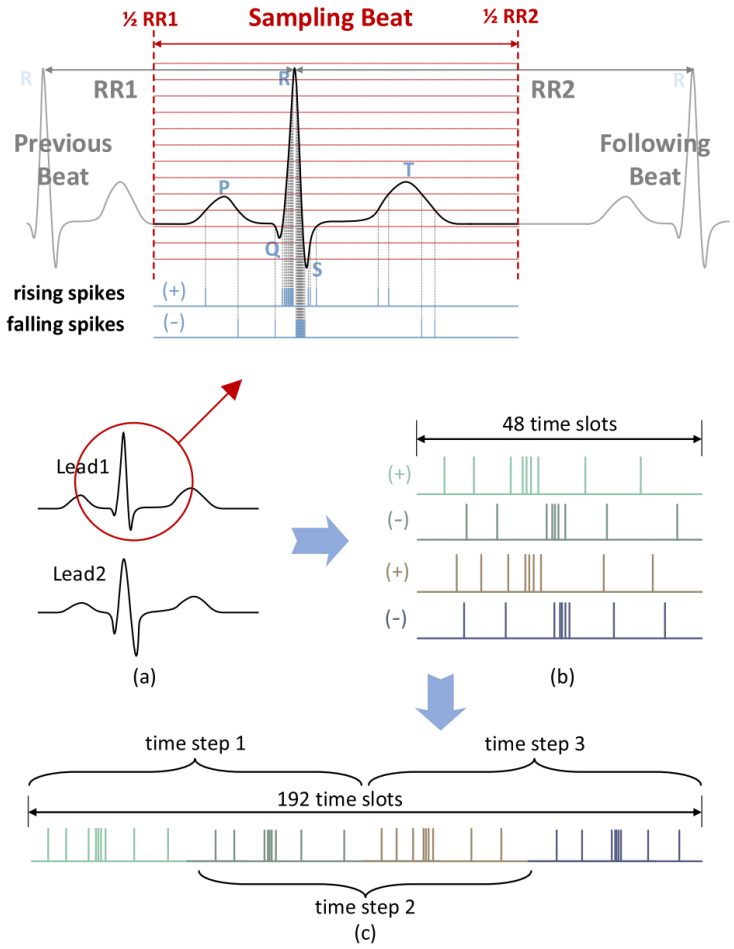


Figure 3.3: LC sampling by Chu et at [14]

The value 0.1 is short in decimal but has a long fraction part in binary. It makes the threshold adjust circuit complex and more resource cost.

Therefore, LC sampling is complex in hardware implementation and has low energy efficiency.

3.4 Summary

In conclusion, Poisson encoding and LC sampling make the calculation time long, leading to much energy consumption. Dual-purpose binary encoding needs to be processed in software because of its complexity, and LC sampling consumes many resources even though it is achievable in hardware. In view of this situation, we hope to develop an encoding scheme that can shorten the calculation time and save resources on hardware while maintaining high accuracy. So, the ECG classification chip can run longer on wearable devices with the same size of battery.

4 Proposed Design

This chapter will explain our contributions from the design aspects in this thesis. The design flow of our proposed work is shown in Figure 4.1; the description order of this chapter will follow the data processing flow. In general, the SNN model is trained on the software, and then the inference process of the trained SNN model is implemented in the hardware. In the software training phase, the data are encoded first, and the encoded data are used for ANN training, and then the ANN model is converted to the SNN model. So, this chapter will introduce the multi-threshold-based encoding scheme, followed by design space exploration where the SNN model is trained, and then the hardware implementation is introduced.

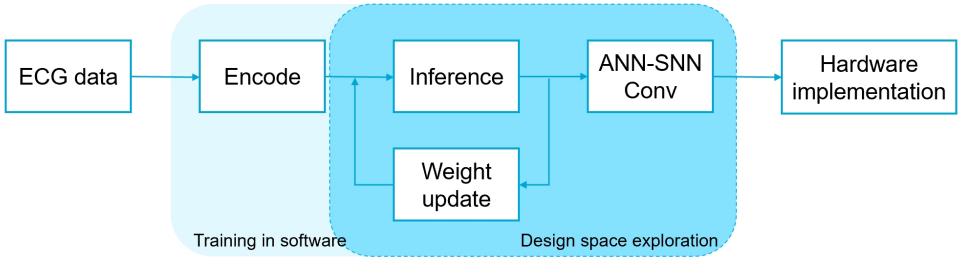


Figure 4.1: The design flow of this thesis work

4.1 Multi-threshold encoding scheme

As we discussed before, the inputs of SNN are spikes form, where we need a specific encoding module to convert the raw data into spikes. Because the ECG signal is continuous, threshold-based encoding is very suitable for it.

For continuous signals, the value of each data point relates to the previous data point. Threshold-based encoding can record the change of this continuous signal. It mainly has two channels. One channel is the incremental channel, and the other is the decremental channel. If the signal increases by a certain value, which is the threshold value, it generates an incremental spike. On the other hand, if the signal decreases by a certain value, a decremental spike is generated. In this way, the change of the continuous signal is recorded. In turn, if we want to resume the ECG signal when there is a spike, it can be achieved by adding or subtracting the threshold from the value of previous data points where there was the last spike. If there is no spike, the value does not change much during this empty time.

Therefore, the threshold-based encoding scheme can generate a small number of spikes, and the encoded data contains both temporal and spatial information. In this work, we proposed the multi-threshold-based encoding scheme that encodes the ECG signal with as few spikes

as possible while maintaining accuracy. Moreover, this encoding scheme is simple to achieve in hardware, which can encode data in real-time.

4.1.1 Analysis of ECG signal

Figure 4.2 shows a randomly selected normal heartbeat from MIT-BIH Arrhythmia database [42]. As we see carefully, between data 80 and 110, the value in QRS peak changes rapidly, and there are small vibrations on the two sides.

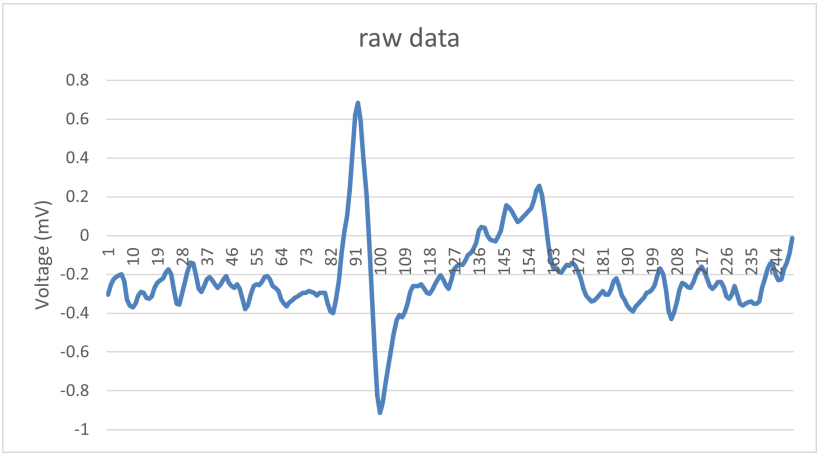


Figure 4.2: A randomly selected normal heartbeat from MIT-BIH Arrhythmia database [42]

If the threshold gap is small, the changes in the ECG signal are recorded. It causes less information loss during the encoding. Meanwhile, for each data point, more bits of data are required to record significant changes in the QRS peak. This leads to more spikes, more storage space, and long computation time, which ultimately leads to higher energy consumption.

If the threshold gap is large, the advantage is that fewer bits are required. However, the vibration on the two sides is ignored to some extent. It results in a lower accuracy. According to our experiment, if the threshold gap is 0.2 mV, there is a sharp drop in the accuracy, which can go down to 90.81%.

4.1.2 Encoding logic

According to the characteristic of ECG signal, the multi-threshold-based encoding is proposed to combine the advantages of the large threshold gap and the small threshold gap. There are two threshold gaps. The large one, Vth_L, is 0.1875 mV; in binary, it is 0.0011 mV. The other smaller one, Vth_S, is 0.0625 mV; in binary, it is 0.0001 mV. The large threshold applies to data between the 60th and 119th, where the QRS peak lies; The small threshold applies to data between the 40th and 229th. As for the data before the 40th and after the 229th, they are discarded.

As shown in Figure 4.3, there will be four channels, inc_L, dec_L, inc_S, and dec_S. Inc_L, dec_L will have 60 bits respectively, and inc_S, dec_S will have 190 bits respectively. The

encoded data is 500 bits in total. Because the number of input neurons is 250, the 500 bits of encoded data will be sent into them in two time-steps. The address mapping is shown in Figure 4.4.

The multi-threshold-based encoding scheme combines the advantage of a large threshold gap and a small threshold gap. It generates a small number of spikes during the QRS peak and has high resolution among small vibrations. Apart from that, the thresholds are 4-bit decimal fractions, making implementation simple in hardware. One comparator is used for one channel, and the precision of comparators is 2^{-4} mV, which is not high. The base value adjust circuit is also simple because the number of bits in the adder is small.

In practical applications, this encoding module can encode ECG data in real-time. After the ADC module samples each ECG data, it can be directly sent to the encoding module for encoding. The encoded data is cached in SRAM, which greatly reduces the amount of data storage by eliminating the need to store the original data.

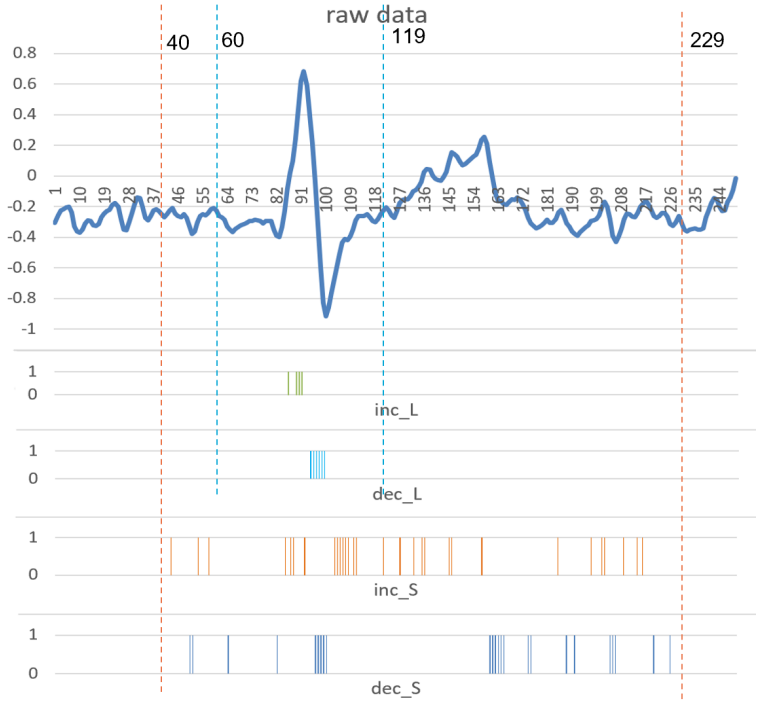


Figure 4.3: The spikes generated by the multi-threshold-based encoding scheme

For this database, the detailed work principle of the multi-threshold-based encoding scheme is as follows:

1. Set the first input data as the initial base value.
2. For data 0-39 and 230-249, no operation will be executed;
3. For data 60-119, the large threshold gap (Vth_L) is applied first; otherwise, go to the next step: If the input data is larger than (base + Vth_L), an incremental spike (inc_L) is generated. If the input data is smaller than (base - Vth_L), a decremental spike (dec_L)

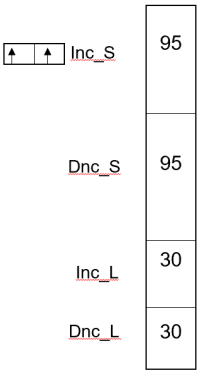


Figure 4.4: The arrangement of the four channels mapping for 250 input neurons of the model

is generated. Then the new base value will be adjusted: (plus sign for inc_L, minus sign for dec_L)

$$base\_value\_new = base\_value\_old \pm Vth\_L \tag{4.1}$$

Then go to Step 4.

4. Within range 40th to 229th, the small threshold gap (Vth_S) works: If the input data is larger than (base + Vth_S), an incremental spike (inc_S) is generated. If the input data is smaller than (base - Vth_S), a decremental spike (dec_S) is generated. Then the new base value will be adjusted: (plus sign for inc_S, minus sign for dec_S)

$$base\_value\_new = base\_value\_old \pm Vth\_S \tag{4.2}$$

5. If any spike is generated, map it to the address as shown in Figure 4.4 according to the channel type and data input sequence.

For wearable devices, there is no need to set the base value at the beginning of each heartbeat. The encoding module can work continuously to detect changes in the ECG. Only some of the required ECG data is kept for caching, while some of the unwanted data is discarded. Furthermore, the cached data is a single-bit instead of an 11-bit fixed-point number, which greatly reduces the amount of storage.

In summary, the multi-threshold-based encoding scheme can reduce the number of spikes while retaining information. This helps to achieve high accuracy and reduce subsequent calculation time, thereby reducing energy consumption. Moreover, this module is simple to implement in the circuit, the resolution of comparators is not high, and the circuit is very resource-saving, eliminating the need for data pre-processing on other software-based chips achieving high energy efficiency.

4.2 Design space exploration

This section aims to acquire an SNN model with good accuracy and suitable architecture for hardware. As discussed in Section 2.2.2, we choose ANN-SNN conversion to train the SNN

model due to its high accuracy and ease of execution features. Firstly, we need to choose a baseline model, then choose a conversion tool, followed by the model training.

4.2.1 Baseline model

A three-stage ANN model [20] is selected as the baseline model. Three states represent three severity levels: normal&mild, moderate, and severe. The architecture of this model is shown in Figure 4.5 with an extra encoding module.

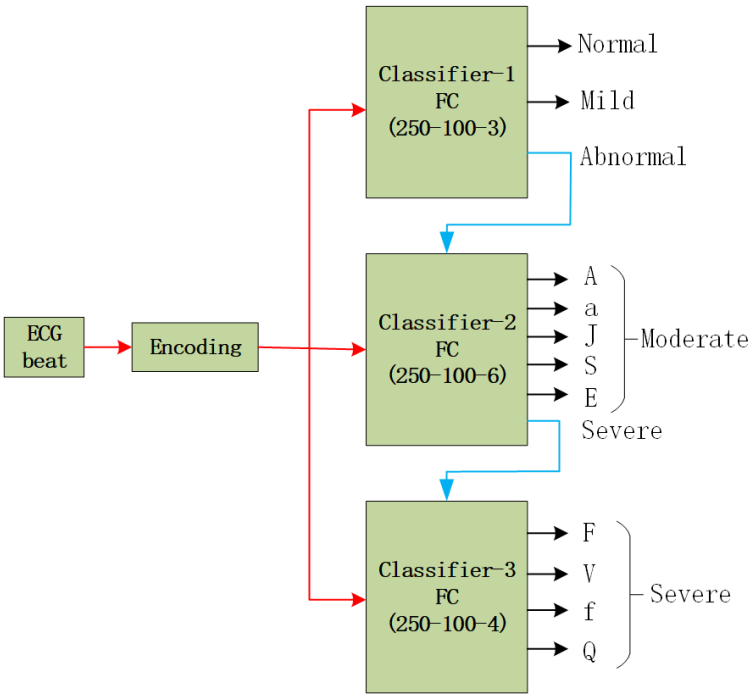


Figure 4.5: The chip architecture with the encoding module and the three-state baseline model (modified from [20])

This baseline ANN model can do 11 types of arrhythmia detection based on MIT-BIH classes [42], while most other ANN models can only do 4 or 5 types of classification based on AAMI classes [52]. More detailed classification not only expands the scope of monitoring but also provides valuable information for doctors to speed up diagnosis during follow-up medical treatment.

This model has high energy efficiency. Most of the types in our daily life and MIT-BIH database are normal and mild, while moderate abnormal and severe types happen very rarely. When the input heartbeat is classified as normal or mild in Classifier-1, inference stops here, and the next two classifiers do not work. Only when the inference type of Classifier-1 is the last type, Abnormal, will this model enter Classifier-2. Similarly, Classifier-3 will only start when Classifier-2 classifies Severe type. Therefore, most of the time, only Classifier-1 is working, and only a small part of the time, the latter two classifiers will be activated.

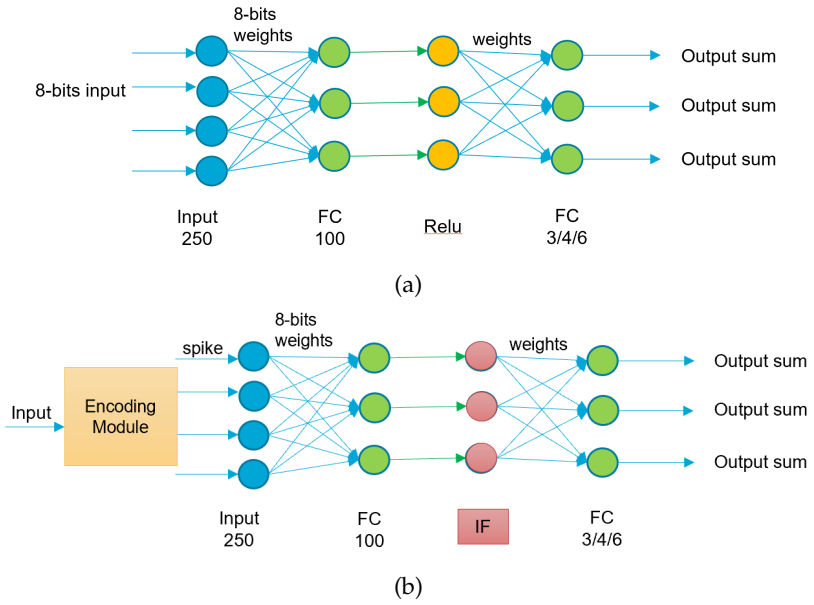


Figure 4.6: The same model implemented in ANN and SNN with 250 input neurons and two fully connected layers. (a) The ANN model (b) The SNN model

Another reason for the high energy efficiency is that these three classifiers share the same architecture, so they can use the same circuit to calculate with different weights. Each classifier has 250 input neurons and two fully connected layers with one ReLu activation layer in between. The first fully connected layer has 100 neurons. The number of neurons in the second fully connected layer depends on its number of types. It could be three, or six, or four. In hardware implementation, the generic classifier can be designed as input250-fc100-fc6. For Classifier-1 and Classifier-3, it is assumed that the extra output neurons are not used.

Figure 4.6a shows the structure of a single classifier. Figure 4.6b is the converted SNN model: ReLu neurons are replaced by IF neurons, and an encoding module is added before the input layer.

Here, an analysis is conducted to compare the resources of these two neural networks. Assume the input is quantized to 8 bits and weights are quantized to 8 bits as well.

Layer	1st FC		2nd FC	
Type of NN	ANN	SNN	ANN	SNN
Multiplier	8*8-bit	\	24*8-bit	\
Adder	24-bit	16-bit	39-bit	15-bit
Sum (bits)	8+8+8=24	8+8=16	24+8+7=39	8+7=15

Table 4.1: The resource estimation based on the computation requirements and comparison between ANN and SNN for the same model

Because the input of the SNN model is either 1 or 0, no multiplier is needed. If the input is 1, add weight; otherwise, add 0, which means no operation.

After this processing, the bit width of the sum of each layer becomes smaller, and the bit width required by the adder also becomes smaller. For the second fully connected layer, the input of ANN is a 24-bit number, while the input of SNN is still 1 bit, so the bit width gap of the sum is even larger. After converting to the SNN model, the bit width of the sum is reduced from 39 bits to 15 bits.

Through this comparison table, we can see that under the same structure, SNN is more energy-efficient than ANN because SNN model calculation does not require a multiplier, requires fewer operations, and the bit width of adders is much smaller.

4.2.2 ANN-SNN Conversion tool

To convert the ANN model to the SNN model, we choose the tool developed by Bu et al[11]. This tool can generate the SNN model with high accuracy and ultra-low latency. High accuracy means the accuracy loss during ANN-SNN conversion is small. They propose a new surrogate function to replace the ReLu function in the activation layer and prove that the conversion loss is close to zero. Ultra-low-latency means the number of time-steps to reach the maximum accuracy is small, which makes the calculation time short and energy consumption low.

We import the ECG database, including the training dataset, validation dataset, and test dataset, the model mentioned in Section 4.2.1, and the encoding function to train an optimum SNN model. Because they have special operations on the activation layer, the ANN model needs to be trained again in their tool, then conversion is executed.

4.2.3 Optimization for accuracy

During the training phase, several techniques are adopted to optimize the accuracy of the SNN model.

- CrossEntropyLoss: Unlike the MNist dataset, the type distribution in the ECG dataset is uneven. Among the types within the classifier, sometimes the severe type takes less proportion. If all the input heartbeats have the same weight in the loss function, during backpropagation, severe type proportion in the loss function is not high. Gradually, in order to improve the overall accuracy, the system pays more attention to optimizing the accuracy of the types that account for large spots. The accuracy of those types with a small proportion is slightly lower. However, the accuracy of minority types is also very important, especially the classification of severe types, which is related to the early warning of ischaemic heart disease or stroke. So CrossEntropyLoss is adopted to adjust the weight in the loss function. More weight is given to the minority types. This measure can improve the local accuracy with sacrificing part of the overall accuracy.
- Scheduler: Training the best model is a process of finding the minimum value of the loss function. The learning rate is related to the step size in the way of finding the minimum loss value. If it is small in the beginning, the model is easy to be stuck in the local optimum solution instead of the global optimum solution. If the learning rate is large in the final stage, it is hard for the model to reach the optimum solution because the adjusting step is too large. Therefore, the variable learning rate is desired, relatively large in the beginning and relatively small in the end. The scheduler can achieve this.

We have tried several different kinds of schedulers, such as LinearLR, MultiStepLR, and CosineAnnealingLR. The effect of them is similar, and CosineAnnealingLR wins over a little bit, so it is used in subsequent training.

4.2.4 Optimization for later hardware design

The weights and biases of the well-trained SNN model are 32-bit float point numbers by default. Calculations with 32-bit float point numbers in hardware implementation are complex. To simplify it, the weights and biases need to be quantized to fixed-point numbers, and the influence on accuracy is expected to be as small as possible.

Because the range of weights is (-2, 2), two bits are needed for the integer part. As for the decimal part, we did some experiments to find the suitable length, of which the result is shown in Table 7.1 in the Appendix. The 8-bit word length, 2 bits in the integer part and 6 bits in the decimal part, is the most suitable quantization length, which has trivial accuracy loss. Quantization shorter than that has obvious accuracy loss.

4.3 Hardware architecture implementation

After the SNN model is determined, the inference calculation is going to be implemented in hardware to achieve a real-time calculation and energy saving. Both the encoding module and classifier module are implemented, so data can be processed directly after sampling from ADC, with no additional software reprocessing.

We follow the flow described in Figure 4.7 to implement the trained and quantized SNN model to the GDSII file that is ready to be manufactured by the foundry.

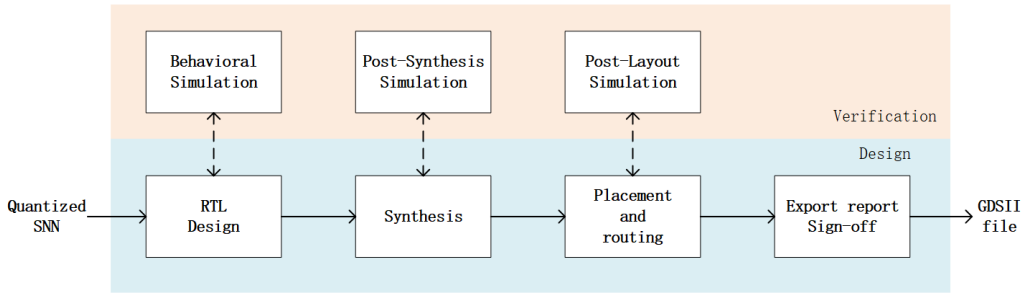


Figure 4.7: The hardware design flow to convert the quantized SNN model to GDSII file

Figure 4.8 is the top-level architecture of the SNN inference chip, Figure 4.9 and Figure 4.10 are the more detailed architecture of the encoding module and classifier module, respectively.

The inputs of the SNN classifier chip are ECG data directly from the sensor, one clock, and one reset. The input data is encoded into spikes and then sent to the neural network classifier module and gets buffered in the memory. When the last data of a heartbeat is encoded, the classifier module will automatically start classification. After the classification

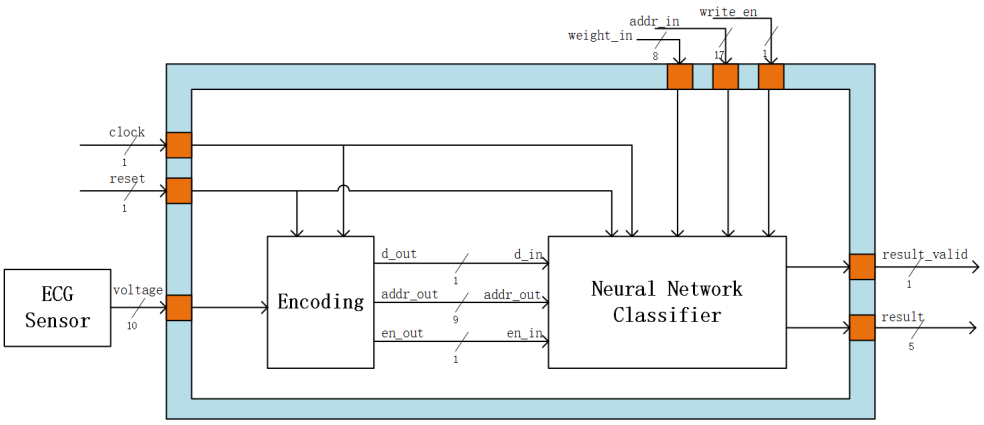


Figure 4.8: The top architecture of the classification chip

is finished, a pulse in the Result_valid port indicates the 5-bit result is ready to be read. The 5-bit result includes a 2-bit number of the classifier and a 3-bit number of the type.

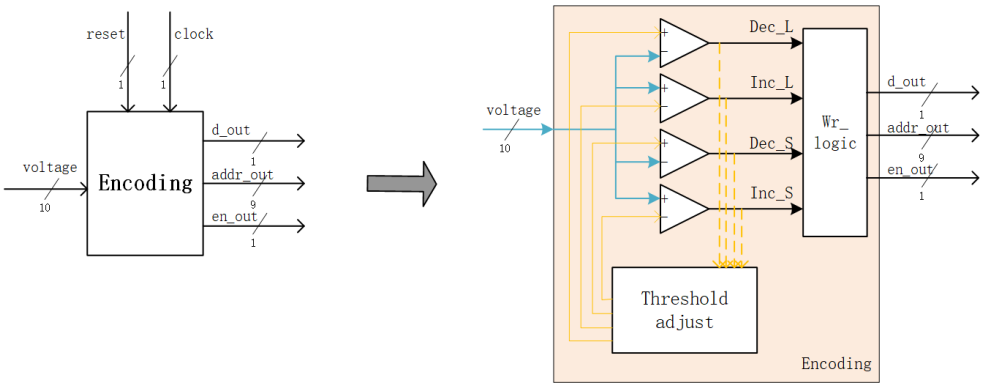


Figure 4.9: The detailed architecture of the encoding module

The encoding module is very simple. There are four comparators, the threshold adjust circuit, and the write logic module. One of the comparator inputs is the data, and the other one is the threshold. Once there is a spike in the output of any comparator, four thresholds are refreshed by the threshold adjust circuit. The encoded data will be sent to the memory classifier module with the address and enable signal. The address and enable signal are generated by the write logic module.

The neural network classifier module can be activated automatically once the encoded data of one heartbeat is all buffered in the memory. This module mainly contains two parts: part one is SRAMs to store weights, data, and interim data, and the other part is the computing unit shared by all three classifiers. The computing unit has one adder that is shared for the two fully connected layers and IF membrane potential update. Because those steps are executed at different times, the adder can be reused to save area and power.

Every time the chip boots up, the first thing to do is load weights from Flash memory to the on-chip SRAM. Since the weights are reloaded every time it boots, the model can also

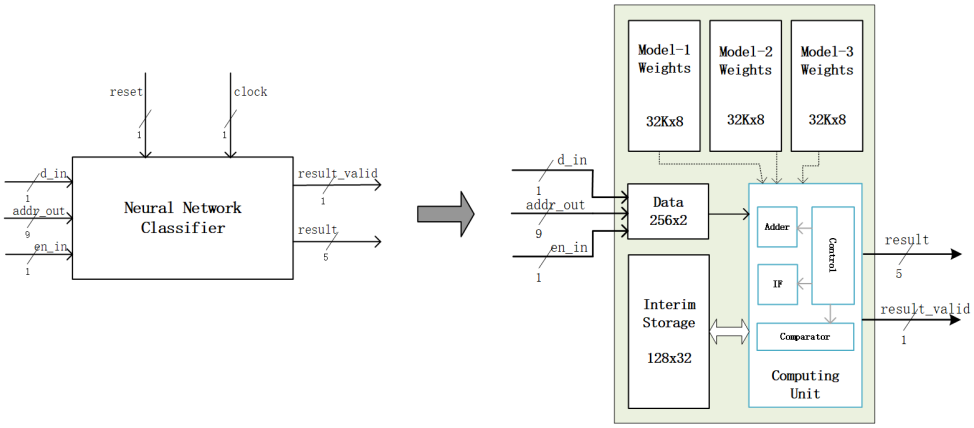


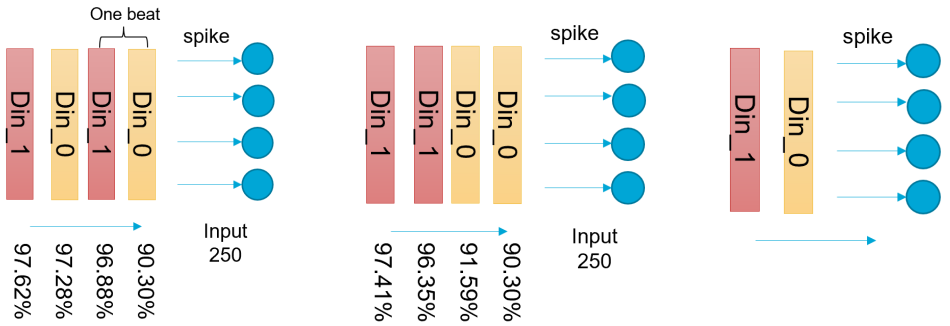
Figure 4.10: The detailed architecture of the neural network classifier module

be customized by training with the client's own ECG signal which can further improve accuracy.

The specific calculation implementation will be explained in detail in the following section.

4.3.1 Hardware-aware optimization design

Even though the conversion tool claims the number of time-steps is small, calculating the heartbeat only once cannot provide the best accuracy. Take one SNN model in Figure 4.11a as an example. If the encoded heartbeat is provided in once, the accuracy is 96.88%; If sending the encoded heartbeat once again, the accuracy is 97.62%, increased by 0.74%. After that, continuing to send the encoded heartbeat data improves very little. So, we decided to use the calculation of sending the heartbeat twice, as shown in Figure 4.11a.



(a) The original calculation sequence

(b) Swith the order of time-step 2 and time-step3

(c) Merge the adjacent same input vector

Figure 4.11: The optimization steps for redundant calculation caused by the conversion tool

Sending the same heartbeat twice causes some redundant calculations and makes the calculation time longer. Assume Din_0 and Din_1 form the encoded data of one heartbeat. In

time-step 1 and time-step 3, the input is Din_0. For the first fully connected layer, the sum is related to the input only; now that the inputs are the same, the sums of the fully connected layer are the same. Meanwhile, the first fully connected layer takes most of the calculation time in the whole flow. If the calculation of these two time-steps of the same input vector can be combined, it will save a lot of calculation time.

Different from ANN calculation, the membrane potential of IF neurons has the memory of previously computed procedures. So, it is not possible to merge discontinuous time steps without changing the final result. The first attempt is to exchange the order of time-step 2 and time-step 3, as shown in Figure 4.11b. The final accuracy does not change much, which means this attempt is acceptable.

Then, we merge time-steps 1 and 2 into one time-step, and merge time-steps 3 and 4 into one time-step, like Figure 4.11c. To maintain the output sums the same as Figure 4.11b after merging time-steps, some special processes need to be done:

- Since two time-steps are merged into one, the sum of the first fully connected layer should be doubled. In digital circuits, multiplying by two can be done by left-shifting one bit.

$$SUM\_fc1\_new = SUM\_fc1 \times 2 = \{SUM\_fc1, 1'b0\} \tag{4.3}$$

- IF neuron fires at most once per time-step before merging. Now, the two time-steps are mixed into one calculation, so the number of IF neuron fires can be 0 , 1, and 2 times. Instead of comparing with V_{th}, now the membrane potential needs to be compared with both V_{th} and $\{V_{th}, 1'b0\}$, which is double V_{th}, to get fire times.

- As for the second fully connected layer, the addition of weights depends on the number of fires in the IF neuron.

  - If the IF neuron[i] does not fire, there is no impact on the output sum.

  - If the IF neuron[i] fires once, add the weight to the final sum [j]:

$$SUM\_output\_new_j = SUM\_output\_previous + weight_{ij} \tag{4.4}$$

  - If the IF neuron[i] fires twice, add the weight to the final sum [j] twice. This process can be simplified in hardware as follows:

$$\begin{aligned} SUM\_output\_new_j &= SUM\_output\_previous + weight_{ij} \times 2 \\ &= SUM\_output\_previous + \{weight_{ij}, 1'b0\} \end{aligned} \tag{4.5}$$

Both Din_0 and Din_1 need to go through the above calculations; adding their output sums together is the final output sum, and the largest type with the largest accumulated sum is the output result of the inference. The output sum and reasoning result of this optimized model are the same as the model in Figure 4.11b because the optimization step only reduces the amount of calculation, but the algorithm remains the same.

In this way, the calculation time is halved at the cost of a little more complex circuit.

4.3.2 Computation flow in hardware

Since the inference calculation flow become simpler, this subsection will introduce the hardware implementation's detailed flow. Figure 4.12 shows the top-level calculation's finite state machine. It contains sub-FSM for FC1 and IF_FC2, which are depicted in Figure 4.13 and Figure 4.14, respectively.

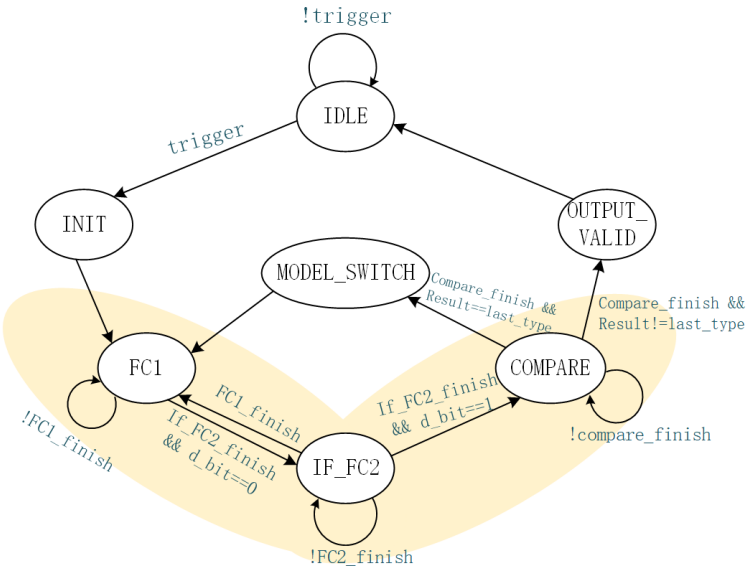


Figure 4.12: The top-level computation flow of SNN inference in hardware

An overview of how the ECG arrhythmia detection on hardware is calculated is provided below:

- The system starts in the IDLE state. It remains in the IDLE state until a trigger pulse occurs. The circuit automatically generates this trigger pulse when the final bit of encoded data is stored in the on-chip SRAM.
- Upon receiving the trigger signal, the circuit initializes the neural network. The membrane potential and the accumulated sum of the second fully connected layer are set to their initial values.
- As there are two time-steps, implying two inputs for input neurons, the network absorbs the first input. This input traverses through the two fully connected layers, yielding a sum. Subsequently, the second input is processed. The accumulated sum of these two inputs is then compared to determine the output neuron with the highest value. This completes one cycle of model computation and yields the result.
- After the calculation of the current classifier is completed, compare all the output sums and the type with the maximum value is the inferred heartbeat type.
- For Classifier-1 and Classifier-2, if the inference result corresponds to the final type within that model, it does not represent the ultimate outcome. Instead, it progresses to the calculation of the subsequent model. The MODEL_SWITCH state is utilized to

facilitate this transition. Furthermore, the input is processed within the same sequence but with distinct weights.

- Once the final result is obtained, the FSM goes to the OUTPUT_VALID state for one cycle before reverting to the IDLE state, awaiting the next heartbeat classification. In the OUTPUT_VALID state, the model number and type number are valid for reading at the output port, indicated by the assertion of a one-bit valid signal.

The above is the top control flow; FC1 and IF_FC2 have more detailed calculation flow, of which their FSM are described in Figure 4.13 and Figure 4.14. On the top level, if the state transits to FC1 from either INIT or MODEL_SWITCH state, it means the new FC1 flow is about to start. There will be an FC1_trigger pulse to activate FC1 FSM, and the top-FSM stays in the FC1 state until the sub-FSM calculation is finished and sends back an FC1_finish pulse. This signal triggers the top FSM transiting from FC1 state to IF_FC2 state, resulting in IF_FC2 sub-FSM starting to work and top FSM stay in this state until the finish signal from the sub-FSM. In this way, the top-FSM and sub-FSMs are switched back and forth to complete the whole computation flow. The following is how the sub-FSM of the first fully connected layer works:

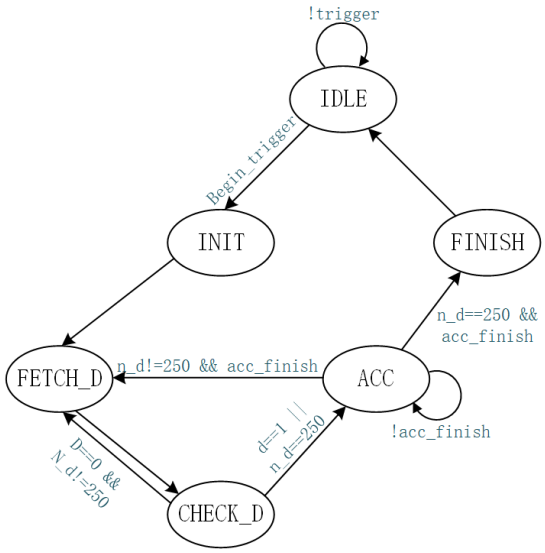


Figure 4.13: The computation flow of the first fully connected layer, which is state FC1 in the top-FSM

- The subsystem begins from the IDLE state; when it receives a trigger pulse from top-FSM, it starts to work.
- The First thing to do is initialize the FC1 calculation system, such as clear the sum that might be left from the previous calculation.
- The bottom triangle is the main calculation part. There are 250 input neurons, which will be checked individually.
- According to the definition of the fully-connected layer, as shown in Equation 4.6, because the input of SNN is either 0 or 1, only ones have an impact on the sum. So,

data will be read from data SRAM one by one, then check whether it is one or zero.

$$sum = \sum weight \times data \tag{4.6}$$

  - If data[i] = 1: all the sum of 1st FC neurons need to be updated. Therefore, the weight of that input neuron and the previous sum of the first hidden layer are read out from memory one by one, then added up, and the new sum will be restored into memory. When all the sum is updated, we need to get the next data[i+1].
  - If data[i] = 0: This data has no impact on the sum of the first hidden layer, so no operation will be executed. We need to check the next data[i+1].
  - after all 250 input neuron's data is checked, the calculation of the first fully connected layer is finished.

- When the calculation process of the first fully connected layer is completed, the sub-FSM goes through one cycle of the FINISH state and returns to the IDLE state.

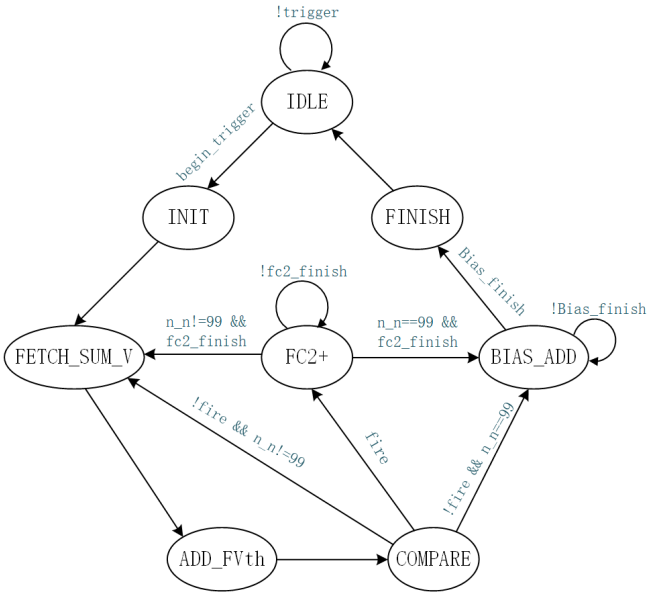


Figure 4.14: The computation flow of IF neurons and the second fully connected layer, which is state IF_FC2 in the top-FSM

In general, state IF_FC2 comprises two major processes in Figure 4.12: IF neuron and the second fully connected layer calculation. The reason to combine them together is for resource saving. If they are addressed in two separate steps, a memory is needed to store the output spike. For the IF neuron, once it fires, this spike is stored in that memory. Then, in the second fully connected layer, the system needs to fetch and check the data one by one, just like FC1 does. It is time-consuming and needs an extra memory.

If these two steps are combined, once there is a spike coming from the IF neuron, it triggers the sum of the second fully connected layer to be updated. The output of the IF is directly

processed without being cached into memory. This saves computation time and reduces circuit resource usage. Here is the description of the sub-FSM of IF+FC2:

- After being triggered, first initialize some related circuits.
- Then fetch the sum of the first FC layer and the membrane potential of the corresponding IF neuron. Add the newly calculated sum to the previous membrane potential to get the new membrane potential.
- Compare the new membrane potential with the threshold of that neuron. If the membrane potential exceeds the V_{th}, that neuron fires and the membrane potential gets soft-reset. As described in Section 4.3.1, add weight or doubled weight to the output sum.
- Store the membrane potential back to memory.
- After all the neurons are cycled in the above way, add bias to the output sum. Then, this sub-FSM sends out a finish pulse and goes back to the IDLE state.

5 Simulation Results

This chapter introduces the software and hardware setup details that are used in this thesis work. Then, the result will be presented, discussed, and compared with related works.

5.1 Simulation setups

Simulation setups include the dataset used in this thesis, software simulation environment, and hardware simulation environment.

5.1.1 Dataset

In this thesis, MIT-BIH Arrhythmia Dataset [42] is adopted for both training and testing. This dataset can be downloaded for free in PysioNet [23]. The data in the database is randomly divided: 60% is used for training, 20% is used for validation during training, and the remaining 20% is used for testing after the model is well-trained.

To judge the effect of the trained model, both accuracy and critical accuracy are used. Accuracy is the proportion of the correct inferred heartbeats among all the tested heartbeats. Critical accuracy refers to the accuracy of the most severe type in the current classifier, which will trigger the work of the next stage (please see Figure 4.5). In Classifier-1, the critical accuracy is the accuracy of the abnormal type. In Classifier-2, the critical accuracy is the accuracy of the server type. As for Classifier-3, there is no critical accuracy because it is the last stage of the whole classifier.

5.1.2 Software Setup

The first step is to train the SNN model in software. Because the newly proposed encoding scheme is not rate encoding, the ANN model training should use the encoded data instead of the original data. Moreover, the conversion tool used in this thesis performs special treatment on ReLu activation neurons so that it has the characteristics of ReLu and reduces the loss of accuracy during the conversion of ANN-SNN. Therefore, it is mandatory to train the ANN model with the new encoding scheme in this tool in the first place.

The training setups are shown in Table 5.1. The model is trained in the PyTorch platform using a conversion tool developed by Bu et al [11]. The baseline ANN model is from Diware et al [20]. During training, stochastic gradient descent (SGD) optimizer, CosineAnnealingLR scheduler, and CrossEntropyLoss are adopted to acquire an optimal ANN model. Then, the well-trained ANN model is converted into an SNN model in the same tool, and the weights are quantized to 8 bits for saving area in hardware implementation.

Parameters	Specification/Source
Weights	8-bit
Baseline model architecture	[Diware-TBioCAS'2023] [20]
ANN-SNN conversion tool	[Bu-ICLR'2022] [11]
Dataset	MIT-BIH ECG arrhythmia dataset [42]
Model training	PyTorch
Optimizer	Stochastic gradient descent (SGD)
scheduler	CosineAnnealingLR

Table 5.1: The software setups for model training

5.1.3 Hardware Setup

After the SNN model is determined, weights, bias, and threshold are exported from the software, now the model is ready to be implemented on hardware. Firstly, the hardware circuit is described in Verilog. Then, the design is synthesized in Genus with TSMC 40 nm technology, 1.1 V V_{dd}, and a 100 MHz clock. After that, the netlist generated by synthesis is implemented into the layout in Innovus. At each step, simulation in QuestaSim with timing is necessary to confirm that the delay will not cause the function to be different from the expected design in the actual application. The circuit is tested in three cases: Worst case (WC), best case (BC), and typical case (TC). The hardware setups are concluded in Table 5.2.

Parameters	Specification
Technology node	TSMC 40 nm
Vdd	1.1V
Clock	100 MHz
PVT corner	Best case (BC) Worst case (WC) Typical case (TC)
EDA tool - Simulation	QuestaSim
EDA tool - Synthesis	Genus
EDA tool - Place and route	Innovus

Table 5.2: The hardware setups for circuit design and implementation

5.2 SNN training

The accuracy and critical accuracy of the SNN model and of the corresponding baseline ANN model are listed in Table 5.3.

The SNN model achieves 97.42% accuracy for classifying normal and mild types of heartbeats, 96.69% accuracy for classifying moderate types of heartbeats, and 96.65% accuracy for severe types of heartbeats. As for the critical accuracy, for Classifier-1, it is 90.07% and 98.60% for Classifier-2. That means the newly proposed multi-threshold-based encoding scheme maintains almost all the ECG information, and the trained SNN model has an excellent effect on heartbeat arrhythmia detection.

Model	Classify type	Accuracy		Critical accuracy	
		Baseline ANN	SNN	Baseline ANN	SNN
Classifier-1	Normal+mild	98.30%	97.42%	95.69%	90.07%
Classifier-2	Moderate	98.30%	96.69%	98.99%	98.60%
Classifier-3	Severe	97.26%	96.65%	N/A	N/A

Table 5.3: The accuracy and critical accuracy of the baseline model [20] and the trained SNN model

Compared with the baseline ANN model, the overall accuracy is close. The average accuracy loss is 1.0%. The accuracy loss results from encoding, conversion, weight quantization, and input data quantization. On classifier-1, the critical accuracy has a relatively large loss, reaching 5.62%. This large accuracy loss is caused by the encoding scheme, according to our experiment. If we use the original data to do training and inference, the critical accuracy is not that large. But this accuracy loss is acceptable because when the heart is unhealthy, the arrhythmia does not happen only once; instead, it repeats. During the occurrence of abnormal heartbeats, as long as abnormal heartbeats can be correctly detected several times, early warning can be provided to the patient, so even a 5.62% loss of critical accuracy on Classifier-1 will not affect its function.

5.3 Hardware implementation

The inference in hardware has the same weights, bias, and threshold as that in software, and the structure of the model is still the same, so the inference result in hardware is the same as that in software. It is verified in simulation that not only the final inference result is the same as that of the software, but even the calculation results of the intermediate steps are the same as the software. So, the accuracy of hardware is identical to the software result shown in Table 5.3. In this section, the result of hardware implementation focuses more on energy consumption.

Since the SNN model is event-driven, the number of input spikes plays a decisive role in energy consumption. The first thing we need to do is count the number of spikes generated after encoding. All heartbeats in the dataset are counted, and the result is shown in Table 5.4.

Heartbeat type	Number	Percentage	Average no. of spikes	Min no. of spikes	Max no. of spikes
Normal+mild	97619	89.19%	53	1	206
Moderate	2887	2.64%	42	15	134
Severe	8946	8.17%	73	2	208
Total	109452	100%	54	\	\

Table 5.4: The spike count after encoding the whole MIT-BIH Arrhythmia database [42]

In the MIT-BIH Arrhythmia Database, 89.19% are normal or mild heartbeats, of which the encoded spikes range from 1 to 206. The average number of spikes per heartbeat is 53. Only 2.64% are moderate abnormal heartbeats, and they are encoded into 42 spikes per heartbeat

on average. 8.17% of the heartbeats in the dataset are severe abnormal types, and the average number of encoded spikes is 73. The overall weighted average is 54 spikes per heartbeat.

Assuming people have 200 heartbeats per minute at most, the time interval for those heartbeats is 300 ms. To achieve real-time processing, the inference time of each heartbeat should be shorter than 300 ms. The longest calculation time in MIT-BIH Arrhythmia Dataset comes from the heartbeat of a severe type, which needs to go through three classifiers and it is encoded into 208 spikes. If the inference result of every stage is correct (it means the inference does not stop at the first two stages due to wrong inference), it takes 680 μs to calculate in hardware, much shorter than 300 ms. It proves that even the most time-consuming inference can meet real-time computing requirements. For heartbeats in all these three categories, the calculation time is greatly shorter than the time interval of one heartbeat, and the slack is pretty sufficient for real-time calculations in a wearable device.

Because the minimum number and the maximum number of spikes deviate too much from the average and are rare cases, they will not be used in the subsequent calculation and analysis. The average values are considered in this work. It can be seen from these statistics that each heartbeat originally contained 250 11-bit fixed-point data, but after encoding, there are only 54 spikes on average, and the encoding compression rate is very high.

Next, the calculation time needs to be obtained. We first find the heartbeats with the average number of encoded spikes in these three categories. Then, perform hardware simulation on them. We find the start time and end time of the classification in the hardware simulation and calculate the inference time of each category. The result of average computation time and average energy consumption is shown in Table 5.5. Because normal or mild heartbeats only go through one stage in the model, the average classification time is the shortest, 71.60 μs, with the clock frequency at 100 MHz. The average calculation for moderate types and severe types are 123.58 μs and 280.20 μs, respectively. Because they go through more classifiers, the calculation time is longer accordingly. By timing the proportion in the dataset, the weighted average calculation time is 90.02 μs. The average energy consumption is obtained in the latter steps.

Heartbeat type	Number	No. of classifiers	Average cal. time (μs)	Average energy consumption (nJ)
Normal+mild	97619	1	71.60	46.03
Moderate	2887	2	123.58	79.45
Severe	8946	3	280.20	180.14
Total/Average	109452		90.02	57.88

Table 5.5: The average computation time and their average energy consumption

Because we do not have SRAM Macros, memories are not implemented in the hardware yet; only computing units of encoding and classifier modules are achieved. After going through synthesis, placement, and routing, the layout is generated, as shown in Table 5.6. Ports connected to memories are extracted to the pin so that SRAM Macros can be added and connected manually in the layout later.

After placement and routing, information about the layout can be reported by Innovus, such as the area, power, setup/hold time slack. Apart from that, the area and power of memories need to be calculated and estimated from the documents provided by the foundry [1] (please see Table 7.2 in Appendix). The area in the layout is measured to be 9409 μm^2, and six SRAMs should take 674832 μm^2 according to TSMC documents. Therefore, the total area

of the chip containing the required SRAMs should be 0.68 mm^2. The area distribution is described in Figure 5.1. The pie chart shows that memories take 98.62% of the area, which is surprisingly large, and only 1.38% is occupied by the computing unit. Among the computing units, the classifier takes up 77.13%, and encoding takes up the remaining area.

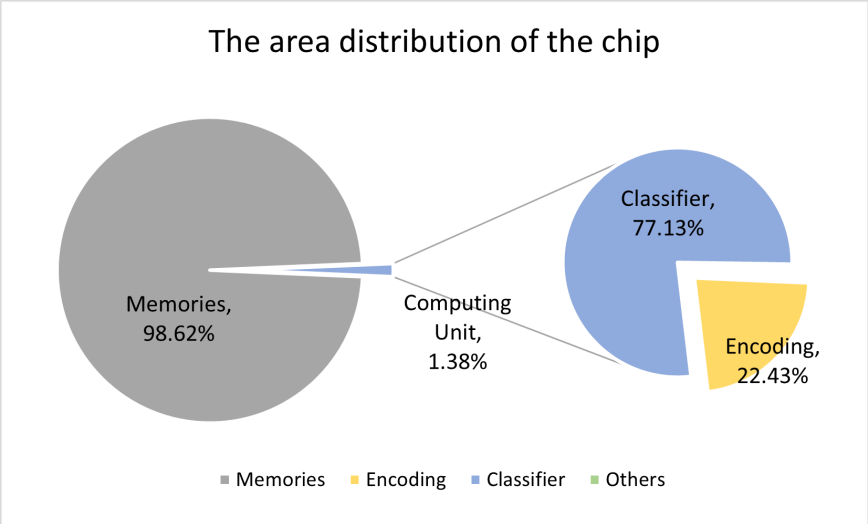


Figure 5.1: The area distribution of the chip in simulation

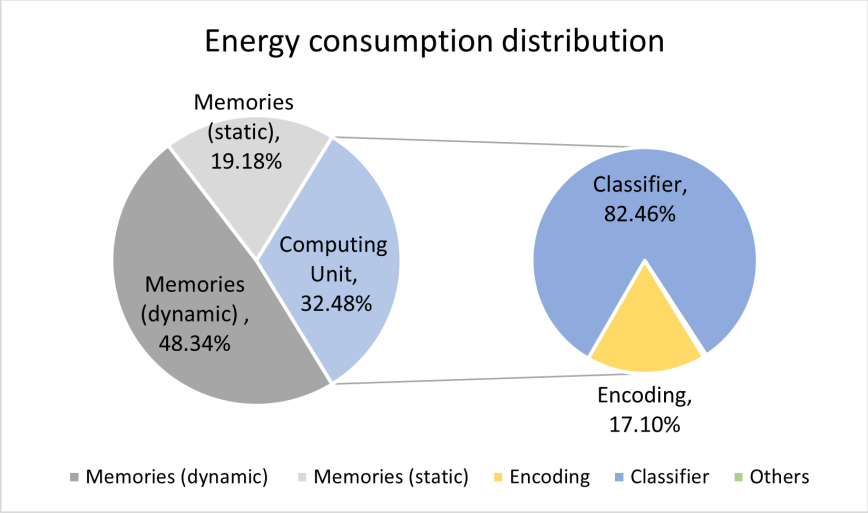
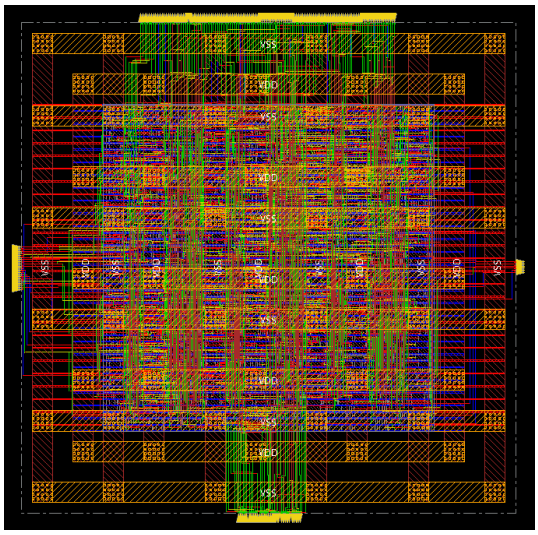


Figure 5.2: The energy consumption distribution of the chip in simulation

The power of memories is also estimated from the document. The quick reference table provides the leakage current, read current, and write current, from which we can approximately calculate the dynamic power and static power. The power reported by Innovus with a more realistic activation factor is 208.83 μW. Combining the power of the computing unit from Innovus, estimated power of memories in Table 7.2 and the average calculation time

from Table 5.5, the average energy consumption per heartbeat classification is estimated to be 57.88 nJ. The energy consumption distribution is shown in Figure 5.2. Since memories occupy more than 98% of the area, it's not surprising that most energy consumption comes from memories, which take up 67.52% of the total energy consumption. Among memories, the dynamic energy consumption is 27.98 nJ, and the static energy consumption is 11.10 nJ. Only 18.80 nJ is caused by the computing unit.

On the left of Table 5.6 is the layout of the computing unit with input ports on the left edge and output ports on the right edge. Pins on the top side and bottom side should be connected to SRAMS.



Process	TSMC 40 nm CMOS
Clock frequency	100 MHz
Area (mm^2)	0.68
Core Area (mm^2)	0.0094
Energy /classification (nJ) (on average)	57.88
Time /classification (μs) (on average)}	90.02
Operation type	Spiking

Table 5.6: The layout of the computing unit and the summary result of the chip after placement and routing

5.4 Comparison with other works in ECG classification

Since the proposed multi-threshold-based encoding and LC sampling proposed in [14] are both threshold-based encoding, Table 5.7 compares these two works in different aspects.

LC sampling needs to use the signal from two leads, while this work only uses the signal from one lead. After encoding, their data amount is 2880 bits in the actual calculation, including 0 and 1, while the data amount of this work is 500 bits in total, and 54 of them are 1. And because we use dynamic threshold gaps, the number of spikes generated by the same heartbeat data will be less than LC sampling, so the calculation time will be shorter and less energy consumption, not to mention that they use more raw data than ours. Besides, their encoding scheme is complex to be implemented into hardware, while ours is simple in hardware because the threshold gap width is short in binary.

In summary, multi-threshold-based encoding has more advantages than LC sampling in energy efficiency and hardware implementation.

	LC sampling [Chu-TBioCAS'2022]	Multi-threshold-based encoding [This work]
	Both are threshold-based encoding	
Number of Lead	Two leads	One lead
Number of encoded data bits (/beat)	2880 bits	500 bits
Threshold gap	Fixed threshold gap	Dynamic threshold gaps
Threshold adjust circuit	complex	simple

Table 5.7: The comparison between LC sampling [14] and multi-threshold-based encoding proposed in this work

This work is compared with four recent ECG classification works. One of them is the baseline ANN model, Diware et al's work, and the remaining three are SNN models. The comparison is listed in Table 5.8. In the table, we can see that the accuracy of those works is very close.

These three SNN models are all classified based on AAMI, so the number of their classifications is only 5 or 4. The baseline ANN model has a more detailed classification, based on the 11 classifications of MIT-BIH, which provided more information for doctors, and this work is also carried out with such a detailed classification.

Diware et al's work is implemented in computing-in-memory (CIM), which has already kept energy consumption per heartbeat very low compared to other digital realizations. When this model is converted to the SNN model and achieved in digital circuits, energy consumption per heartbeat gets further lower, approximately 52.73% of the ANN model. It shows that the SNN model is more energy efficient than the traditional ANN model.

The four SNN works in this table adopt different encoding schemes and they have different model structures. Benefiting from the encoding scheme with very few spikes, the excellent three-layer classifier structure, and the sophisticated circuit design, this work consumes less energy consumption than other works. Not to mention that their encoding is done off-chip, and the extra consumption of this part is not counted.

	[Diware-TBioCAS'2023] [20]	[Chu-TBioCAS'2022] [14]	[Mao-TBioCAS'2022] [41]	[Amirshahi-TBioCAS'2019] [6]	This work
Technology	——	40 nm	28 nm	——	40 nm
Computing unit type	CIM	Digital	Digital	Analogue	Digital
Neural network type	ANN	SNN	SNN	SNN	SNN
Spike encoding method	——	LC sampling	Dual-purpose binary encoding	Poisson encoding	Multi-threshold-based encoding
Learning rule	——	STBP	ANN-SNN Conv	R-STDP	ANN-SNN Conv
Accuracy	98.29%	98.22%	97.36%	97.9%	97.42%
Chip area(mm^2)	0.11	0.3246	0.54	——	0.68
Energy /classification (μJ)	0.11	20.51 - 0.75	0.3	1.78	0.058 (on average)
Number of classification	11	5	5	4	11

Table 5.8: The comparison of the state-of-the-arts with our work

6 Conclusion

This chapter concludes the thesis and gives some suggestions for future work.

6.1 Conclusion

To reduce deaths from cardiovascular disease, in this thesis, the ECG classifier on the hardware is designed for wearable devices to detect cardiac arrhythmia in real-time. Patients can catch abnormal signals at an early stage and seek medical treatment in time to avoid worsening or severe attacks of cardiovascular disease. The Spiking Neural Network (SNN) is adopted in this work for automatic feature extraction and inference for its low energy consumption feature.

Since the inputs of SNN are spikes, the encoding module is required to transfer the original data into spikes. Considering the importance of the encoding effect in the final result and the shortcomings of existing encoding schemes, we propose a new encoding scheme, Multi-threshold-based encoding. This new encoding scheme encodes the ECG data from 250 numbers to 54 spikes on average. Meanwhile, this new encoding scheme is easy to achieve in hardware with a small area, avoiding data preprocessing on another chip and realizing continuous processing on the same chip, which is helpful for energy saving.

The training method of the SNN model is ANN-SNN conversion because of the high accuracy and the easy way to achieve it. We have selected the three-state classifier from Diware et al [20] and the ANN-SNN conversion tool from Bu et al [11] to train the SNN model. After adjusting hyperparameters and doing some training optimizations, the obtained SNN model has 97.42% accuracy. Except for the 5% loss of the critical accuracy of Classifier-1, most of the other accuracy is very high and close to the baseline ANN model, indicating that the proposed encoding scheme retains most of the information of the ECG signal.

The inference and encoding modules are implemented into hardware through RTL design, simulation, synthesis, placement, and routing. As we have analyzed, SNN is simpler in hardware compared with traditional ANN because it does not need a multiplier, and the width of the calculation result is smaller. With sophisticated designs in this thesis to reuse the calculation unit, improve memory utilization, and compress calculation time, the hardware chip achieves an average calculation time of 90.02 μs and 57.88 nJ energy consumption per classification. The hardware contains both the encoding module and the inference module. Because the number of operations of SNN in hardware is related to the number of input spikes, this work also shows a suitable encoding scheme is essential to the SNN model for accuracy, calculation time, and energy consumption.

6.2 Future works

- ANN-SNN conversion is a convenient way to train SNN model because the training method of ANN is very common now, and the conversion way avoids complex surrogate functions in SNN backpropagation. Even though Bu et al's tool [11] declares the time-steps are shortest compared to other tools, in this thesis, the inputs need to be sent in the model at least twice for higher accuracy. Therefore, it would be better if the conversion tool could be further improved by reducing the time-steps to shorten the calculation time.
- From the result of the layout, memories take up most of the energy consumption. If energy consumption is expected to be lower, memory should be the main point, either reducing the memory power or reducing the number of reading and writing operations.

7 Appendix

7.1 The table of weight quantization

WL (word length)	FL (decimal length)	Classifier-1 ANN Acc(%)	Classifier-2 ANN Acc(%)	Classifier-3 ANN Acc(%)
32-bit float number		97.51	97.08	97.21
10	8	97.50	97.04	97.15
9	7	97.49	97.00	96.93
8	**6**	**97.44**	**96.91**	**97.04**
7	5	97.36	97.04	96.53
6	4	97.07	96.62	95.36

Table 7.1: The accuracy and critical accuracy of different widths of weights

7.2 The table of area and power estimation of SRAMs

	Function of the SRAM	Area (μm^2)	$I_{leakage}$ (μA/MHz)	I_{read} (μA/MHz)	I_{write} (μA/MHz)	$I_{dynamic}$ (μA/MHz)	Leakage power (μW)	Dynamic power (μW)	Power (μW)
1	Weight of classifier-1	221191.74	36.84	6.60	0.00	6.60	40.52	145.22	185.75
2	Weight of classifier-2	221191.74	36.84	0.00	0.00	0.00	40.52	0.00	40.52
3	Weight of classifier-3	221191.74	36.84	0.00	0.00	0.00	40.52	0.00	40.52
4	Data	2293.89	0.34	0.67	0.62	1.29	0.38	28.40	28.78
5	FC1_sum	4481.61	0.62	1.40	1.72	3.12	0.68	68.57	69.25
6	Membrane potential	4481.61	0.62	1.40	1.72	3.12	0.68	68.57	69.25
Total		674832.31					123.31	310.77	434.08

Table 7.2: The area and power estimation according to the information provided by TSMC [1]

Bibliography

[1] Tsn40lp2prf_20071100_130a_quick_reference_table.pdf.

[2] https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death, .

[3] https://world-heart-federation.org/wp-content/uploads/World-Heart-Vision-2030.pdf, .

[4] https://world-heart-federation.org/resource/cardiovascular-disease-infographic/.

[5] Linear/Fully-Connected Layers User's Guide. URL https://docs.nvidia.com/deeplearning/performance/dl-performance-fully-connected/index.html.

[6] A. Amirshahi and M. Hashemi. Ecg classification algorithm based on stdp and r-stdp neural networks for real-time monitoring on ultra low-power personal wearable devices. *IEEE Transactions on Biomedical Circuits and Systems*, 13(6):1483–1493, 2019. doi: 10.1109/TBCAS.2019.2948920.

[7] A. Antony, S. R. Paulson, and D. J. Moni. Asynchronous Adaptive Threshold Level Crossing ADC for Wearable ECG Sensors. *Journal of Medical Systems*, 43(3):78, Feb. 2019. ISSN 1573-689X. doi: 10.1007/s10916-019-1186-8. URL https://doi.org/10.1007/s10916-019-1186-8.

[8] D. E. Becker. Fundamentals of electrocardiography interpretation. *Anesth. Prog.*, 53(2): 53–63; quiz 64, 2006.

[9] G.-q. Bi and M.-m. Poo. Synaptic modification by correlated activity: Hebb's postulate revisited. *Annual Review of Neuroscience*, 24(1):139–166, 2001. doi: 10.1146/annurev.neuro.24.1.139. URL https://doi.org/10.1146/annurev.neuro.24.1.139. PMID: 11283308.

[10] M. Bouvier, A. Valentian, T. Mesquida, F. Rummens, M. Reyboz, E. Vianello, and E. Beigne. Spiking Neural Networks Hardware Implementations and Challenges: A Survey. *ACM Journal on Emerging Technologies in Computing Systems*, 15(2):22:1–22:35, Apr. 2019. ISSN 1550-4832. doi: 10.1145/3304103. URL https://dl.acm.org/doi/10.1145/3304103.

[11] T. Bu, W. Fang, J. Ding, P. Dai, Z. Yu, and T. Huang. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks, 2023.

[12] S. Chauhan and L. Vig. Anomaly detection in ecg time signals via deep long short-term memory networks. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–7, 2015. doi: 10.1109/DSAA.2015.7344872.

[13] S. S. Chowdhury, N. Rathi, and K. Roy. One timestep is all you need: Training spiking neural networks with ultra low latency, 2021.

[14] H. Chu, Y. Yan, L. Gan, H. Jia, L. Qian, Y. Huan, L. Zheng, and Z. Zou. A neuromorphic processing system with spike-driven snn processor for wearable ecg classification. *IEEE Transactions on Biomedical Circuits and Systems*, 16(4):511–523, 2022. doi: 10.1109/TBCAS.2022.3189364.

[15] C. J. Deepu, X. Xu, X. Zou, L. Yao, and Y. Lian. An ecg-on-chip for wearable cardiac monitoring devices. In *2010 Fifth IEEE International Symposium on Electronic Design, Test Applications*, pages 225–228, 2010. doi: 10.1109/DELTA.2010.43.

[16] L. Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[17] S. Deng and S. Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=FZ1oTwcXchK.

[18] P. U. Diehl and M. Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Front. Comput. Neurosci.*, 9:99, Aug. 2015.

[19] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2015. doi: 10.1109/IJCNN.2015.7280696.

[20] S. Diware, S. Dash, A. Gebregiorgis, R. V. Joshi, C. Strydis, S. Hamdioui, and R. Bishnoi. Severity-based hierarchical ecg classification using neural networks. *IEEE Transactions on Biomedical Circuits and Systems*, 17(1):77–91, 2023. doi: 10.1109/TBCAS.2023.3242683.

[21] C. Frenkel, M. Lefebvre, J.-D. Legat, and D. Bol. A 0.086-mm^2 12.7-pj/sop 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm cmos. *IEEE Transactions on Biomedical Circuits and Systems*, 13(1):145–158, 2019. doi: 10.1109/TBCAS.2018.2880425.

[22] W. Gerstner, R. Ritz, and J. L. van Hemmen. Why spikes? hebbian learning and retrieval of time-resolved excitation patterns. *Biol. Cybern.*, 69(5-6):503–515, Sept. 1993.

[23] A. Goldberger, L. Amaral, L. Glass, J. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C. K. Peng, and H. E. Stanley. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *Circulation*, 101(23):E215–20, June 2000.

[24] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama. Neural Coding in Spiking Neural Networks: A Comparative Study for Robust Neuromorphic Systems. *Frontiers in Neuroscience*, 15, 2021. ISSN 1662-453X. URL https://www.frontiersin.org/articles/10.3389/fnins.2021.638474.

[25] B. Han and K. Roy. Deep Spiking Neural Network: Energy Efficiency Through Time Based Coding. In A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, pages 388–404, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58607-2. doi: 10.1007/978-3-030-58607-2_23.

[26] A. Y. Hannun, P. Rajpurkar, M. Haghpanahi, G. H. Tison, C. Bourn, M. P. Turakhia, and A. Y. Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature Medicine*, 25(1):65–69, Jan. 2019. ISSN 1546-170X. doi: 10.1038/s41591-018-0268-3. URL https://doi.org/10.1038/s41591-018-0268-3.

[27] Y. He and J. Zhao. Temporal convolutional networks for anomaly detection in time series. *Journal of Physics: Conference Series*, 1213(4):042050, jun 2019. doi: 10.1088/1742-6596/1213/4/042050. URL https://dx.doi.org/10.1088/1742-6596/1213/4/042050.

[28] A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.*, 117(4):500–544, Aug. 1952.

[29] E. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572, 2003. doi: 10.1109/TNN.2003.820440.

[30] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.12.005. URL https://www.sciencedirect.com/science/article/pii/S0893608017302903.

[31] J. Kim, H. Kim, S. Huh, J. Lee, and K. Choi. Deep neural networks with weighted spikes. *Neurocomputing*, 311:373–386, 2018. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.05.087. URL https://www.sciencedirect.com/science/article/pii/S0925231218306726.

[32] S. Kiranyaz, T. Ince, and M. Gabbouj. Real-time patient-specific ecg classification by 1-d convolutional neural networks. *IEEE Transactions on Biomedical Engineering*, 63(3):664–675, 2016. doi: 10.1109/TBME.2015.2468589.

[33] L. Lapicque. Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. 1907.

[34] C. Lee, S. S. Sarwar, P. Panda, G. Srinivasan, and K. Roy. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14, 2020. ISSN 1662-453X. doi: 10.3389/fnins.2020.00119. URL https://www.frontiersin.org/articles/10.3389/fnins.2020.00119.

[35] C. Li, Z. Shang, L. Shi, W. Gao, and S. Zhang. Ic-snn: Optimal ann2snn conversion at low latency. *Mathematics*, 11(1), 2023. ISSN 2227-7390. doi: 10.3390/math11010058. URL https://www.mdpi.com/2227-7390/11/1/58.

[36] T. Li and M. Zhou. Ecg classification using wavelet packet entropy and random forests. *Entropy*, 18(8), 2016. ISSN 1099-4300. doi: 10.3390/e18080285. URL https://www.mdpi.com/1099-4300/18/8/285.

[37] L. Lilly and H. M. School. *Pathophysiology of Heart Disease: A Collaborative Project of Medical Students and Faculty*. Online access: Lippincott LWW Health Library: Integrated Basic Sciences Collection. Wolters Kluwer, 2016. ISBN 9781451192759. URL https://books.google.nl/books?id=6_a5oQEACAAJ.

[38] F. Liu, W. Zhao, Z. Wang, X. Yang, and L. Jiang. Simsnn: A weight-agnostic reram-based search-in-memory engine for snn acceleration. In *2023 Design, Automation Test in Europe Conference Exhibition (DATE)*, pages 1–2, 2023. doi: 10.23919/DATE56975.2023.10136973.

[39] Y. Liu, Z. Wang, W. He, L. Shen, Y. Zhang, P. Chen, M. Wu, H. Zhang, P. Zhou, J. Liu, G. Sun, J. Ru, L. Ye, and R. Huang. An 82nw 0.53pj/sop clock-free spiking neural network with 40µs latency for alot wake-up functions using ultimate-event-driven bionic architecture and computing-in-memory technique. In *2022 IEEE International Solid- State Circuits Conference (ISSCC)*, volume 65, pages 372–374, 2022. doi: 10.1109/ISSCC42614.2022.9731795.

[40] R. Lozano, M. Naghavi, K. Foreman, et al. Global and regional mortality from 235 causes of death for 20 age groups in 1990 and 2010: a systematic analysis for the global burden of disease study 2010. *The Lancet*, 380(9859):2095–2128, 2012. ISSN 0140-6736.

[41] R. Mao, S. Li, Z. Zhang, Z. Xia, J. Xiao, Z. Zhu, J. Liu, W. Shan, L. Chang, and J. Zhou. An ultra-energy-efficient and high accuracy ecg classification processor with snn inference assisted by on-chip ann learning. *IEEE Transactions on Biomedical Circuits and Systems*, 16(5):832–841, 2022. doi: 10.1109/TBCAS.2022.3185720.

[42] G. Moody and R. Mark. The impact of the mit-bih arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001. doi: 10.1109/51.932724.

[43] E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. doi: 10.1109/MSP.2019.2931595.

[44] S. Padmavathi and E. Ramanujam. Naïve bayes classifier for ecg abnormalities using multivariate maximal time series motif. *Procedia Computer Science*, 47:222–228, 2015. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2015.03.201. URL https://www.sciencedirect.com/science/article/pii/S187705091500469X. Graph Algorithms, High Performance Implementations and Its Applications (ICGHIA 2014).

[45] S. Park, S. Kim, B. Na, and S. Yoon. T2fsnn: Deep spiking neural networks with time-to-first-spike coding. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2020. doi: 10.1109/DAC18072.2020.9218689.

[46] B. Rajendran, A. Sebastian, M. Schmuker, N. Srinivasa, and E. Eleftheriou. Low-power neuromorphic hardware for signal processing applications: A review of architectural and system-level design approaches. *IEEE Signal Processing Magazine*, 36(6):97–110, 2019. doi: 10.1109/MSP.2019.2933719.

[47] V. Randazzo, J. Ferretti, and E. Pasero. Ecg watch: a real time wireless wearable ecg. In *2019 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pages 1–6, 2019. doi: 10.1109/MeMeA.2019.8802210.

[48] S. Saadatnejad, M. Oveisi, and M. Hashemi. Lstm-based ecg classification for continuous monitoring on personal wearable devices. *IEEE Journal of Biomedical and Health Informatics*, 24(2):515–523, 2020. doi: 10.1109/JBHI.2019.2911367.

[49] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in Neuroscience*, 13, 2019. ISSN 1662-453X. doi: 10.3389/fnins.2019.00095. URL https://www.frontiersin.org/articles/10.3389/fnins.2019.00095.

[50] J. Sjöström and W. Gerstner. Spike-timing dependent plasticity. *Scholarpedia*, 5(2):1362, 2010. doi: 10.4249/scholarpedia.1362. revision #184913.

[51] R. Srivastva. *Ecg Pattern Analysis And Classification For Human Recognition*. PhD thesis, Dr. A.P.J. Abdul Kalam Technical University, 2021. URL http://hdl.handle.net/10603/347775.

[52] A. f. t. A. o. M. I. A. M. Subcommittee. *Recommended Practice for Testing and Reporting Performance Results of Ventricular Arrhythmia Detection Algorithms (proposed)*. AAMI recommended practice. Association for the Advancement of Medical Instrumentation, 1986. URL https://books.google.nl/books?id=ojkGHAAACAAJ.

[53] R. Thilagavathy, R. Srivatsan, S. Sreekarun, D. Sudeshna, P. L. Priya, and B. Venkataramani. Real-time ecg signal feature extraction and classification using support vector machine. In *2020 International Conference on Contemporary Computing and Applications (IC3A)*, pages 44–48, 2020. doi: 10.1109/IC3A48958.2020.233266.

[54] S. Thorpe, D. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381(6582):520–522, June 1996. ISSN 1476-4687. doi: 10.1038/381520a0. URL https://www.nature.com/articles/381520a0. Number: 6582 Publisher: Nature Publishing Group.

[55] N. Wang, J. Zhou, G. Dai, J. Huang, and Y. Xie. Energy-efficient intelligent ecg monitoring for wearable devices. *IEEE Transactions on Biomedical Circuits and Systems*, 13(5): 1112–1121, 2019. doi: 10.1109/TBCAS.2019.2930215.

[56] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi. Direct training for spiking neural networks: Faster, larger, better. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1311–1318, Jul. 2019. doi: 10.1609/aaai.v33i01.33011311. URL https://ojs.aaai.org/index.php/AAAI/article/view/3929.

[57] Z. Yan, J. Zhou, and W.-F. Wong. Energy efficient ECG classification with spiking neural network. *Biomedical Signal Processing and Control*, 63:102170, Jan. 2021. ISSN 1746-8094. doi: 10.1016/j.bspc.2020.102170.

[58] F. Zenke and S. Ganguli. SuperSpike: Supervised Learning in Multilayer Spiking Neural Networks. *Neural Computation*, 30(6):1514–1541, 06 2018. ISSN 0899-7667. doi: 10.1162/neco_a_01086. URL https://doi.org/10.1162/neco_a_01086.